

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DTIC QUALITY INSPECTED 2



THESIS

**DETECTION AND ESTIMATION OF
FREQUENCY HOPPING SIGNALS
USING
WAVELET TRANSFORMS**

by

Howard F. Overdyk

September, 1997

Thesis Advisor:

Co-Advisor:

Monique P. Fargues

Ralph Hippenstiel

Approved for public release; distribution is unlimited.

19980317 067

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE September 1997	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE DETECTION AND ESTIMATION OF FREQUENCY HOPPING SIGNALS USING WAVELET TRANSFORMS			5. FUNDING NUMBERS	
6. AUTHOR(S) Overdyk, Howard F.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (maximum 200 words) <p>This thesis investigates the use of wavelet transforms in the detection and estimation of spread spectrum frequency hopping signals. The technique developed in this work makes only two basic assumptions of a minimum hopping time and a minimum frequency hopping differential. The approach is based on the phase information of the temporal correlation function and the resulting discrete wavelet transform is used to estimate the hopping time of frequency hopping signals. Results show the proposed scheme is robust to additive white noise for SNR levels of 3 dB and above.</p>				
14. SUBJECT TERMS Spread spectrum, Frequency Hopping, Wavelets			15. NUMBER OF PAGES 113	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified		20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

Approved for public release; distribution is unlimited

**DETECTION AND ESTIMATION OF
FREQUENCY HOPPING SIGNALS
USING
WAVELET TRANSFORMS**

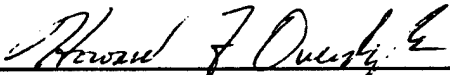
Howard F. Overdyk
Captain, United States Marine Corps
B.S., Iowa State University, 1991


Submitted in partial fulfillment of the
requirements for the degree of

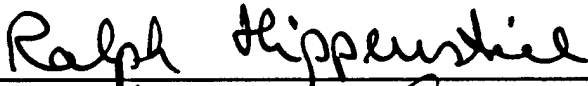
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

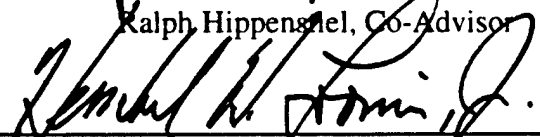
from the

**NAVAL POSTGRADUATE SCHOOL
September 1997**

Author: 
Howard F. Overdyk

Approved by: 
Monique P. Fargues, Thesis Advisor


Ralph Hippenskiel, Co-Advisor


Herschel H. Loomis, Jr., Chairman
Department of Electrical and Computer Engineering

ABSTRACT

This thesis investigates the use of wavelet transforms in the detection and estimation of spread spectrum frequency hopping signals. The technique developed in this work makes only two basic assumptions of a minimum hopping time and a minimum frequency hopping differential. The approach is based on the phase information of the temporal correlation function and the resulting discrete wavelet transform is used to estimate the hopping time of frequency hopping signals. Results show the proposed scheme is robust to additive white noise for SNR levels of 3 dB and above.

TABLE OF CONTENTS

I. INTRODUCTION.....	1
A. BACKGROUND.....	1
B. GOALS.....	2
II. FREQUENCY HOPPING SIGNALS.....	5
A. SPREADING THE SPECTRUM.....	5
B. FREQUENCY HOPPING EXAMPLE.....	7
C. BENEFITS OF SPREAD SPECTRUM.....	8
1. Interference Suppression.....	8
2. Energy Density Reduction.....	10
3. Multiple Access.....	11
III. WAVELETS.....	13
A. FOURIER ANALYSIS.....	13
1. Fourier Series.....	13
a. Continuous-Time Representation.....	13
b. Discrete-Time Representation.....	15
2. Fourier Transform.....	15
a. Continuous-Time Representation.....	15
b. Discrete-Time Representation.....	16
3. Short-Time Fourier Transform.....	16
B. WAVELET ANALYSIS.....	21
1. Wavelet Series.....	22
a. Continuous-Time Representation.....	22

b. Discrete-Time Representation.....	27
2. Wavelet Transform.....	31
a. Continuous-Time Representation.....	31
b. Discrete-Time Representation.....	32
IV. WAVELET BASED TRANSIENT DETECTION.....	35
A. DETECTING DISCONTINUITIES USING WAVELETS.....	35
1. The Continuous Wavelet Transform.....	37
2. The Discrete Wavelet Transform.....	38
3. The Continuous Wavelet Transform Applied to Functions with Additive Noise.....	41
4. The Discrete Wavelet Transform Applied to Functions with AWGN.....	42
5. Averaging of Scales.....	43
B. THE TEMPORAL CORRELATION FUNCTION.....	47
C. PREPROCESSING TECHNIQUES.....	51
1. Unwrapping the Phase.....	52
2. Median Filter.....	53
3. Differentiation.....	54
V. DETECTION AND ESTIMATION ALGORITHM AND SIMULATION.....	57
A. DETECTION AND ESTIMATION ALGORITHM.....	57
B. SIMULATION.....	59
1. Assumptions.....	59
2. Signal Generation.....	60
C. APPLICATION OF DETECTION AND ESTIMATION ALGORITHM TO A SAMPLE FRAME.....	61

1. Choosing a Sample Frame.....	61
2. The Temporal Correlation Function.....	61
3. Extracting Phase Information from the TCF.....	62
4. Constructing the Pulse.....	63
5. Detecting Discontinuities with the DWT.....	65
6. Constructing the Detection Vector.....	66
7. Threshold Determination.....	67
D. RESULTS.....	69
1. Detection.....	69
2. Estimation.....	70
VI. CONCLUSIONS AND RECOMMENDATIONS.....	73
A. CONCLUSIONS.....	73
B. RECOMMENDATIONS.....	74
APPENDIX A. MATLAB SOURCE CODE.....	75
APPENDIX B. THRESHOLD DETERMINATION.....	93
LIST OF REFERENCES.....	99
INITIAL DISTRIBUTION LIST.....	101

I. INTRODUCTION

A. BACKGROUND

Spread spectrum communications schemes have received ever increasing attention over the past two decades as numerous civilian applications have joined the military applications [1]. Originally, spread-spectrum communications were developed to provide a secure means of communication in the hostile environments of World War II [1]. This work remained mostly classified until 1970s, but by the late 1970s the literature was starting to amass [1]. Naturally, it was not long before civilian applications for spread spectrum communications began to develop. One such system, the Global Positioning System (GPS), which was originally developed for the military, soon found widespread civilian application in position location for civilian vehicles, commercial vehicles and ships, and hunters and fishermen [1]. Other applications include cellular telephony and personal communication systems, both of which are still growing rapidly [1].

With this increased use of spread-spectrum communications, naturally, came the question of unauthorized interception. This question was not only of importance to the military, but also to communications regulatory boards, such as the Federal Communications Commission (FCC) [2]. This question has, therefore, given rise to a considerable amount of research over the past decade focused on the detection and estimation of spread-spectrum communications signals [2]-[6]. Two main assumptions typically found in the literature are that the hop timing is constant and known, and that the hopping frequencies are selected from a known class of candidate frequencies. Even

when the hop timing is not assumed known, it is still usually assumed constant [3]. These assumptions generally restrict the detection and estimation schemes to *frequency hopping* (FH), one of the more popular spread-spectrum communications techniques.

During approximately the same period as the detection and estimation of FH signals research was being conducted an analysis technique called *wavelet analysis* began finding wide spread use in signal processing [7]. One reason wavelet analysis was of particular interest is its multiresolution analysis capabilities. In other words, wavelet analysis allows one to “zoom in” for a detailed look at a signals characteristics, or “zoom out” for a global view of the signal. To use a map analogy, the former would be a map of city streets (a “large” scale map) while the latter would be a map of the United States (a “small” scale map).

A spread-spectrum communications signal consists of both long duration relatively low frequency components (“large” scale) in the carrier frequencies and short duration high frequency components (“small” scale) in the transitional hops. As mentioned above, wavelet analysis is, in theory, well suited for analyzing signals of this type. Indeed, the two have met [4] and wavelet analysis was shown effective in detecting frequency hopping signals.

B. GOALS

The primary goal of this thesis is to provide a new approach for the detection and estimation of frequency hopping signals which makes none of the restrictive assumptions listed above. By not making such restrictive assumptions, it is hoped that a secondary

goal of wider application to the detection and estimation of other spread-spectrum communications techniques (i.e., direct sequencing, time hopping and hybrids of the three) given in Peterson [1] can be obtained.

This thesis is composed of six chapters with this introduction being the first. Chapter II introduces frequency hopping signals which are derived from one particular spread-spectrum communications signaling technique. Chapter III defines and explains wavelet analysis by analogy with the more familiar Fourier analysis. Chapter IV provides the theory behind wavelet-based transient detection and introduces a few preprocessing tools which will be used in the detection and estimation algorithm. Chapter V enumerates the steps of the algorithm, describes the simulation used in the testing of the algorithm, applies the algorithm to an example FH signal, and provides the results of the simulation. Chapter VI provides a summary, conclusions and proposed further study.

II. FREQUENCY HOPPING SIGNALS

In communications there exist techniques for spreading the spectrum of transmitted signals. These techniques are called *spread spectrum* (SS) because the actual transmission bandwidth is **much** greater than the minimum bandwidth which would otherwise be required to transmit the given information [8]. Such techniques provide several benefits which include signal interference suppression, low probability of signal detection, and multiple access by many signals to the same spectral region. This section introduces frequency hopping (FH) signals which are one candidate scheme in SS applications. We first describe the concept of frequency hopping, then give a simple example, and finally, discuss some of the aforementioned benefits of spectrum spreading.

A. SPREADING THE SPECTRUM

It has been shown that for signals of bandwidth W and duration T that the dimensionality of the signaling space is approximately equal to $2WT$ [9]. The idea behind SS is to increase this dimensionality (i.e., to increase the size of the signal space), in order to realize the benefits mentioned above. It can readily be seen that there are only two options for accomplishing this spread, either increase W or increase T . One method for increasing the bandwidth, W , is called frequency hopping. Increasing the time variable, T , can be accomplished via time hopping (TH). A detailed survey of other SS techniques and/or TH techniques can be found in Sklar [8].

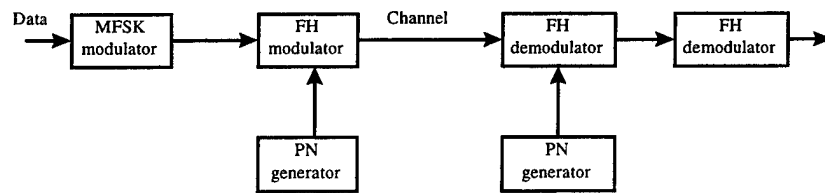


Figure 2.1: FH/MFSK system

Although the particular modulation scheme is not of great importance to the detection algorithm described in this thesis, M-ary frequency shift keying (MFSK) will be used in the discussion and example which follow since this modulation scheme is most commonly found in FH systems. It should be noted, however, that it will be the FH properties of the signal, not the particular modulation scheme which will be of importance later when the detection algorithm is described. In FH systems the available bandwidth, W_{ss} , is subdivided into a large number of frequency slots, N . The data symbol modulates the carrier frequency, f_k , where k is selected by a pseudorandom number (PN) generator to be between 1 and N . The FH system can be thought of as a two-step modulation process, although in practice the two steps would be combined into one. The first step, MFSK modulation, would be followed by the second step, FH modulation, as shown in Figure 2.1. Note that the pattern of transmitted signal in the time-frequency plane will be affected primarily by the FH modulation step, regardless of the particular modulation scheme or the number of symbols transmitted per hop. For a given hop, the occupied bandwidth, W , is the same as that for conventional MFSK. However, averaged

over many hops a much greater bandwidth equal to W_{ss} is occupied, and the spectrum has been spread.

B. FREQUENCY HOPPING EXAMPLE

Assume that the bandwidth, W_{ss} , is equal to 20 MHz (5 MHz - 25 MHz) and that the minimum frequency hopping interval, Δf , is equal to 1 KHz. This implies that $f_1=5$ MHz, $f_2=5.001$ MHz, $f_3=5.002$ MHz, ..., $f_N=25$ MHz where:

$$N = \frac{W_{ss}}{\Delta f}, \quad (2.1)$$

equals 20,000. Now, if the PN generator produces the sequence {7500, 1250, 17727, 11667, 2143}, the first hopping frequency will be:

$$f_{7500} = 5,000,000 + \Delta f \cdot 7,500 = 12.5 \text{ MHz}, \quad (2.2)$$

and similarly, the next four frequencies will be $f_{1250} = 6.25$ MHz, $f_{17727} = 22.727$ MHz, $f_{11667} = 16.667$ MHz, and $f_{2143} = 7.143$ MHz. The pattern for this particular FH signal in the time-frequency plane is shown in Figure 2.2. For a given hop (for example, between 0 and 20 μ s) we see that the occupied bandwidth, W , is the same as that for conventional MFSK. However, averaged over many hops, a bandwidth of W_{ss} is occupied. Hence, the spectrum has been spread from W to W_{ss} . Note that depending on the definition of bandwidth that this example does not completely cover W_{ss} . However, it should be obvious that as transmission time increases and the carrier frequency, f_k , for each hop is randomly selected, eventually, the entire W_{ss} will be covered.

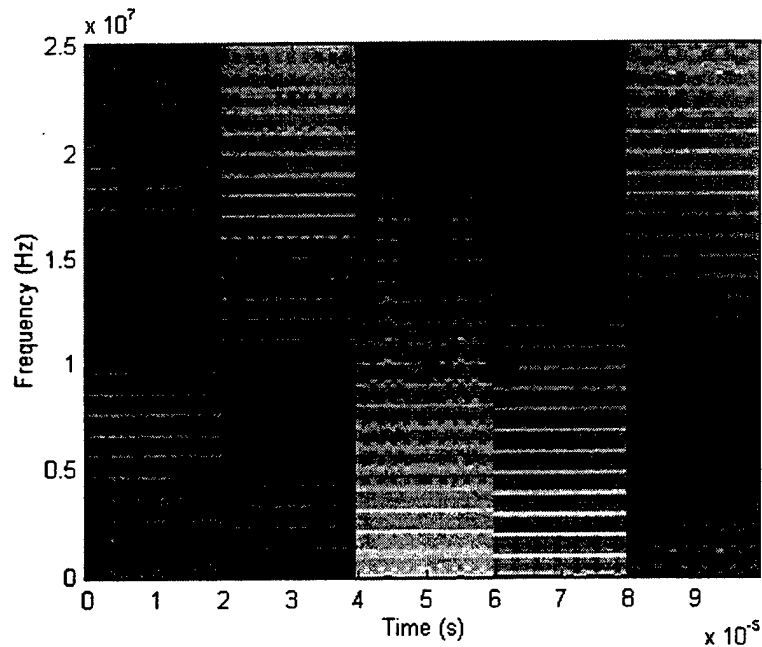


Figure 2.2: An example of a FH pattern

C. BENEFITS OF SPREAD SPECTRUM

1. Interference Suppression

Consider signals of bandwidth W_{ss} and duration T where W_{ss} is the SS bandwidth. By definition, white Gaussian noise (WGN) has infinite power spread uniformly over all frequencies [8]. It can be seen from this definition that regardless of where within W_{ss} the actual information resides, it suffers from the same amount of interference from WGN. However, jammers must be of a fixed *finite* power. Therefore, the jammer must decide to either spread this power in small quantities over the whole or a large portion of

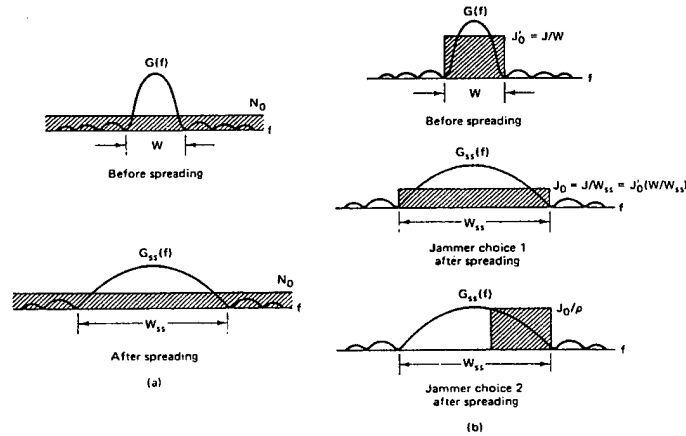


Figure 2.3: Effect of spectrum spreading (a) in the presence of GWN and (b) in the presence of an intentional jammer. From Ref. [8]

W_{ss} or to concentrate this finite power on one or a few smaller portions of the total bandwidth, W_{ss} .

Figure 2.3 shows the effects, in the frequency domain, of spreading the spectrum both in the presence of GWN and an intentional jammer. In Figure 2.3a, the shaded region, N_0 , is the power spectral density (PSD) of GWN, $G(f)$ is the power spectrum of a signal $g(t)$ before spreading it from signal space W to W_{ss} , and $G_{ss}(f)$ is the power spectrum of $g(t)$ after spreading its signal space. We can see that interference due to GWN has the same effect on the signal both before and after the spectrum spreading, since the GWN's *infinite* power is spread uniformly over all frequencies. In Figure 2.3b, J is the *finite* power of the jammer and the shaded region, $J'_0 = J/W$, is the jammer's power spectral density. Here we see, in contrast to the GWN case, that SS does reduce the interference due to an intentional jammer, since it typically will not occupy the same spectral location as the jammer for the duration of the transmission. Once the spectrum is

spread, the jammer has the choice of either covering the entire spread spectrum bandwidth, W_{ss} , lightly with its *finite* power as shown in Figure 2.3b (middle), or of only covering only a portion of W_{ss} with increased concentration of its *finite* power as shown in Figure 2.3b (bottom). Note how the shaded region which is the PSD, J_0 , of the jamming signal is, in the case of Figure 2.3b (middle), wide and short, indicating that the effective jamming power is distributed relatively lightly over **all** of W_{ss} . In the case of Figure 2.3b (bottom), however, the shaded region is narrower, but taller, indicating that the effective power being delivered is increased by reducing the spread spectrum signal bandwidth coverage to, ρ ($0 \leq \rho \leq 1$), a **portion** of W_{ss} . Therefore, the jammer has chosen to concentrate its *finite* power, J , to a portion of the spread spectrum, W_{ss} , in order to have an effect on the SS signal when it resides in this portion (the shaded region of Figure 2.3b (bottom)).

2. Energy Density Reduction

Another benefit of spread spectrum techniques in signaling is that they provide a *low probability of detection* (LPD) [8]. As seen earlier a SS signal's energy is spread throughout a bandwidth much larger than that used in conventional schemes. Therefore, the signal energy present at any given frequency is very small. Hence, the SS signal will appear buried in noise to any potential receiver which does not possess the synchronized spreading signal for FH demodulation.

Energy density reduction also is desirable in unrelated and broader (i.e., civilian as well as military) applications. In satellite communications, the downlink transmissions

must meet international regulations on the spectral density which is transmitted [8]. Spread spectrum schemes allow an increase in the total transmitted power while adhering to these regulations.

3. Multiple Access

A benefit of spread spectrum signaling which has found wide application in cellular telephony is *code-division multiple access* (CDMA) [1]. This technique provides simultaneous users with separate unique spreading signal codes. Therefore, many simultaneous users can share the same spectrum, W_{ss} , by simply not occupying the same portion of the spectrum at the same time. The insurance that they will not occupy the same coordinates in W_{ss} is provided by the *unique* spreading signal code. In addition, CDMA offers some measure of privacy as it will be difficult to decode the signal without a synchronized replica of the spreading signal code.

In this chapter we have introduced *frequency hopping* signals. Particular attention was paid to the FH properties of the signal instead of the particular modulation scheme, since these are the properties which will be used to detect and estimate hopping times. In the following chapter *wavelets* and *wavelet transforms* are introduced. The wavelet transform will be the primary tool used in our detection and estimation algorithm developed in Chapter V.

III. WAVELETS

Wavelet analysis has become widely known within the last ten to twelve years, although similar analysis techniques were employed in various disciplines within engineering, physics, and mathematics as early as the beginning of the century [7]. There are various methods for describing and explaining wavelet analysis. However, the most useful method might be that based on analogies with the more familiar classical Fourier analysis [7, 10, 11]. Thus, first, we will provide a brief review of Fourier analysis, and next present an introduction to wavelet analysis.

A. FOURIER ANALYSIS

1. Fourier Series

a. *Continuous-Time Representation*

Recall that a periodic function, $x(t)$, with a fundamental period T_0 and fundamental frequency $f_0 = \frac{1}{T_0}$, has a complex Fourier series expansion in terms of complex exponentials given by:

$$x(t) = \sum_{k=-\infty}^{+\infty} a_k e^{jk2\pi f_0 t}, \quad (3.1)$$

where $e^{jk2\pi f_0 t}$ are the set of harmonically related complex exponentials.

The Fourier series coefficient, a_k , represents the amount the k -th harmonic of the basis functions, $e^{jk2\pi f_0 t}$, is present in $x(t)$. Multiplying both sides of Equation 3.1 by $e^{-jn2\pi f_0 t}$ and then integrating the result over one period T_0 , leads to:

$$\int_0^{T_0} x(t) e^{-jn2\pi f_0 t} dt = \sum_{k=-\infty}^{+\infty} a_k \left[\int_0^{T_0} e^{j(k-n)2\pi f_0 t} dt \right]. \quad (3.2)$$

Note that $\int_0^{T_0} e^{j(k-n)2\pi f_0 t} dt = 0$ for $k \neq n$, as the integration is carried out over one full period of the complex exponential. Thus:

$$\begin{aligned} \int_0^{T_0} x(t) e^{-jn2\pi f_0 t} dt &= \sum_{k=-\infty}^{+\infty} a_k \left[\int_0^{T_0} e^{j(k-n)2\pi f_0 t} dt \right] \\ &= a_n T_0. \end{aligned} \quad (3.3)$$

Therefore, Equation 3.3 leads to:

$$a_n = \frac{1}{T_0} \int_{T_0} x(t) e^{-jn2\pi f_0 t} dt. \quad (3.4)$$

Equations 3.1 and 3.4 define the Fourier series of a periodic signal. Recall that Equation 3.1 is called the *synthesis* equation and Equation 3.4 the *analysis* equation [12]. Note that the a_0 coefficient is given as:

$$a_0 = \frac{1}{T_0} \int_{T_0} x(t) dt, \quad (3.5)$$

represents the dc component, or average value, of $x(t)$.

b. Discrete-Time Representation

For discrete time periodic applications, the discrete time Fourier series expansion of a periodic signal, $x[n]$, is given by:

$$x[n] = \sum_{k=0}^{N-1} a_k e^{jk(2\pi/N)n} , \quad (3.6)$$

where the Fourier series coefficient, a_k , is given by:

$$a_k = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-jk(2\pi/N)n} . \quad (3.7)$$

2. Fourier Transform

a. Continuous-Time Representation

The introduction of the Fourier transform which allows the representation of not only periodic (as we saw in the last section), but also *aperiodic* signals as linear combinations of complex exponentials was one of Fourier's most significant contributions to this form of signal analysis [12].

The Fourier transform is obtained by examining the limiting behavior of the Fourier series representation as the period, T_0 , is allowed to grow arbitrarily large to infinity [12]. The resulting *Fourier transform* pair is given by:

$$x(t) = \int_{-\infty}^{+\infty} X(f) e^{j2\pi ft} df , \quad (3.8)$$

$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi ft} dt . \quad (3.9)$$

By convention, the analysis equation, Equation 3.9, is called the *Fourier transform*, while the synthesis equation, Equation 3.8, is called the *inverse Fourier transform* [12]. Note that the Fourier series coefficients can be calculated via the *Fourier transform* since, for periodic signals, $X(f)$ will have amplitudes $\{a_k\}$ as given by Equation 3.4 which will occur only at a discrete set of harmonically related frequencies, kf , $k=0, \pm 1, \pm 2, \dots$. In contrast, for aperiodic signals the complex exponentials of Equation 3.9 occur with amplitude $X(f)(df)$ over a continuum of frequencies [12].

b. Discrete-Time Representation

Fourier synthesis and analysis equations can also be defined for discrete time aperiodic signals using a discrete-time representation of the Fourier transform. Letting $\Omega = 2\pi f$ represent the digital frequency to distinguish from the continuous frequency f , leads to:

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{j\Omega n} d\Omega, \quad (3.10)$$

$$X(\Omega) = \sum_{n=-\infty}^{+\infty} x[n] e^{-j\Omega n}. \quad (3.11)$$

3. Short-Time Fourier Transform

While the Fourier transform works very well for stationary signals, its limitations quickly become apparent when this analysis technique is applied to a non-stationary signal. Consider the signal from the example in section II-B. The signal $x_1(t)$ consists of

one frequency, $f_{7500} = 12.5$ MHz, for $0 \leq t \leq 20 \mu\text{s}$ and then a hop to frequency, $f_{1250} = 6.25$ MHz, for $20 \mu\text{s} \leq t \leq 40 \mu\text{s}$, as shown in Figure 3.1a. The Fourier transform $X_1(f)$ clearly shows the two component frequencies of this signal, as illustrated in Figure 3.1c. Now, consider a second signal, $x_2(t)$, of a duration equal to $40 \mu\text{s}$. However, instead of a having a frequency hop the signal is now made up of the summation of two pure sinusoidal signals $x_{7500}(t)$ and $x_{1250}(t)$ with frequencies, f_{7500} and f_{1250} , respectively, each with duration of $40 \mu\text{s}$, as shown in Figure 3.1b. The resulting Fourier transform $X_2(f)$ is very similar to $X_1(f)$, as shown in Figure 3.1d. Thus, this example illustrates the fact that the Fourier transform does not provide any temporal information.

Suppose, now, that we window $x_1(t)$ into two, $20 \mu\text{s}$ sections, $x_{1a}(t)$ and $x_{1b}(t)$. These two new signals will now consist of only one frequency each, f_{7500} and f_{1250} , respectively. If we now take the Fourier transform of $x_{1a}(t)$ and $x_{1b}(t)$, separately, we obtain $X_{1a}(f)$ and $X_{1b}(f)$ as shown in Figure 3.2. Results show that for $0 \leq t \leq 20 \mu\text{s}$, only f_{7500} is present, while for $20 \mu\text{s} \leq t \leq 40 \mu\text{s}$, only f_{1250} is present. Hence, we now have reintroduced some temporal information into our analysis by windowing the data. This, time localization via windowing, is the basic idea behind the short-time Fourier transform (STFT) which provides a surface as the time-frequency structure which represents a given signal in the time-frequency plane [13].

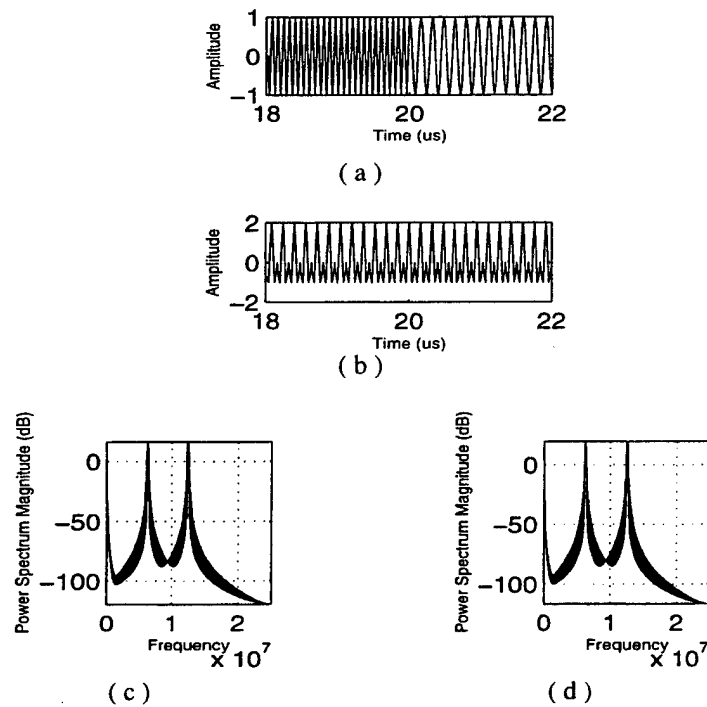


Figure 3.1: Time traces (a) of $x_1(t)$ showing the change from f_{7500} to f_{1250} at $t = 20 \mu s$ and (b) of $x_2(t)$ showing the addition of two single sinusoid signals $x_{7500}(t)$ and $x_{1250}(t)$ over the same time span 18 to $22 \mu s$. The power spectrums of $x_1(t)$ and $x_2(t)$ are shown in (c) and (d), respectively, as being the same even though the two signals are different. All frequencies are given in Hertz.

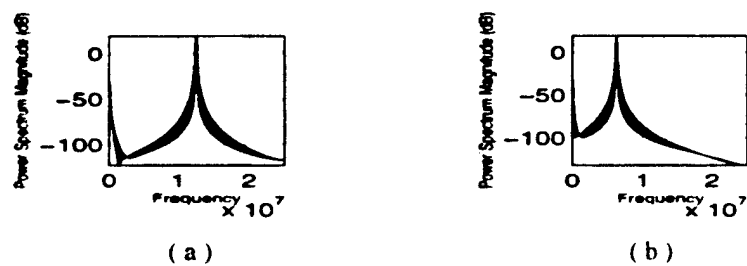


Figure 3.2: Power spectrums (a) of $x_{1a}(t)$ which is the portion of $x_1(t)$ from 0 to $20 \mu s$, and (b) of $x_{1b}(t)$ which is the portion of $x_1(t)$ from $20 \mu s$ to $40 \mu s$ showing how temporal information can be reintroduced into the signal analysis. Frequencies are in Hertz.

Assume that a signal, $x(t)$, is stationary in a window, $g(t)$, centered at time τ . The Fourier transform of the windowed signal $x(t)g^*(t-\tau)$ leads to the STFT expansion given by:

$$\text{STFT}(\tau, f) = \int x(t)g^*(t-\tau) e^{-j2\pi ft} dt, \quad (3.12)$$

where $*$ denotes the complex conjugate. The STFT is a function of two variables, τ and f , which represent time and frequency, respectively. Therefore, the STFT function maps a function, $x(t)$, of only one variable t , representing time, into a two-dimensional time-frequency plane.

There is a dual interpretation of the STFT as shown in Figure 3.3 [7]. The first interpretation is that of “windowing the signal” as illustrated by the sliding window, $g(t)$, in Figure 3.3. In this interpretation, a window size is selected around some center time, τ ,

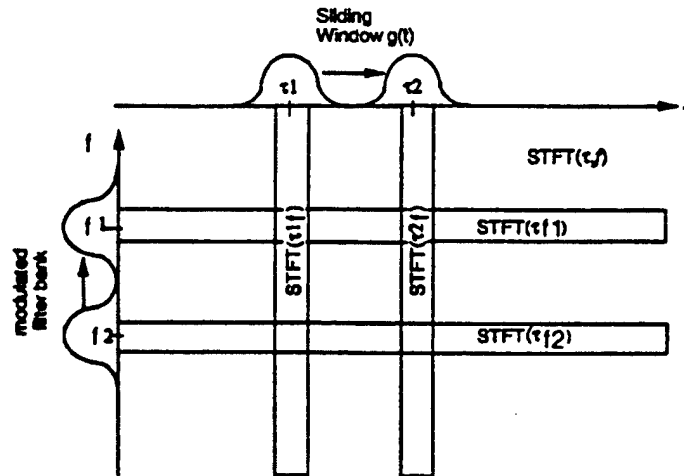


Figure 3.3: The time-frequency plane of the Short-Time Fourier Transform. It can be viewed either as a succession of Fourier Transforms of windowed segments of the signal (vertical bars) or as a modulated filter bank (horizontal bars). From Ref. [7]

and all frequencies within this window are computed. The second interpretation is that of a modulated window function, or a “filter bank” as illustrated by the implied modulated filter bank of Figure 3.3. Under this interpretation a bandpass filter is centered on some frequency, f , and all times within this filter are computed. With this dual interpretation in mind, let us call the bandwidth of the bandpass filter Δf and the window, $g(t)$, size Δt . Now, from the uncertainty principle we recall that the time-bandwidth product, $\Delta t \Delta f$, has a lower bound given by:

$$\Delta t \Delta f \geq \frac{1}{4\pi}. \quad (3.13)$$

Therefore, one cannot simultaneously obtain good time and frequency resolution, but has to trade one for the other. One potential drawback of the STFT is that this time frequency partitioning is **fixed** for the entire transform, as shown in Figure 3.4. Figure 3.4b shows

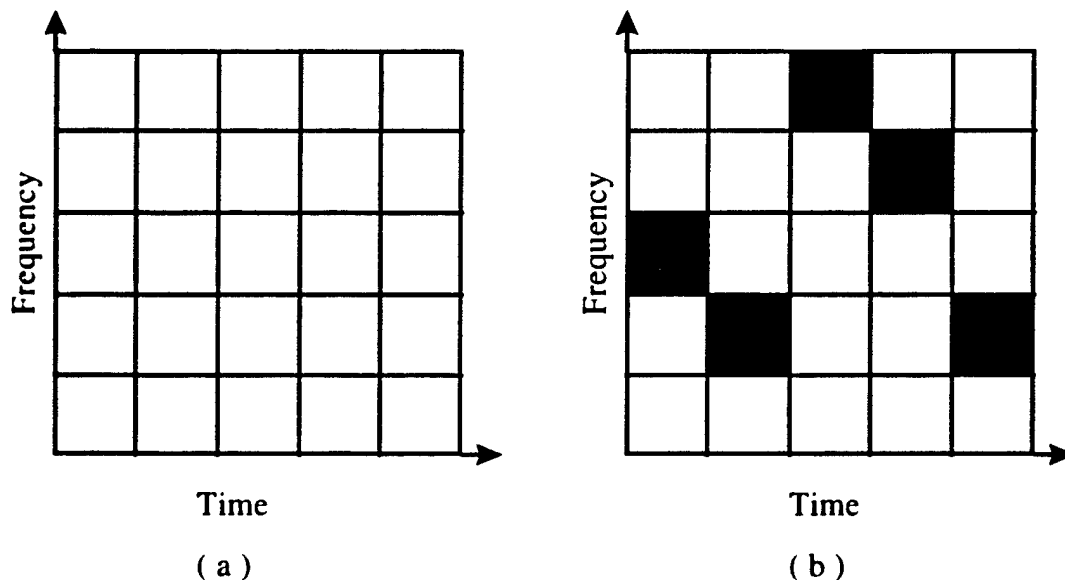


Figure 3.4: Tiling of the time-frequency plane by the Short-Time Fourier Transform (a) a generic case and (b) the pattern from the example in Figure 2.2.

how the tiles are filled in the case of the example given in section II-B. If our signal of interest was composed entirely of transients or entirely of sinusoidal terms we would be able to choose a short window in the former case or a narrow filter in the latter case, and the STFT would prove to be adequate for our analysis. In practice, however, many signals have both long duration sinusoidal and transient components. For these types of signals the STFT is often not the right tool for adequate analysis. What is needed for these types of signals is a multiresolution analysis, one type of which may be obtained with wavelets.

B. WAVELET ANALYSIS

The Fourier series analysis equation, given in Equation 3.4, shows that the coefficients are formed by projecting the signal $x(t)$ onto a set of complex exponential basis functions, $e^{-jk2\pi f_0 t}$, where $k = 0, \pm 1, \pm 2, \dots$. One might call this form of analysis, “wave” analysis, since complex exponentials are wave functions (i.e., sines and cosines). In “wavelet” analysis, instead of using wave functions as our basis, we use wavelets. A *wavelet*, literally meaning “small wave” [14], has its energy concentrated in time [11]. Therefore, although it still has oscillating, wave-like, characteristics facilitating frequency analysis, its localization in time allows transient analysis [11].

1. Wavelet Series

This section is titled “wavelet series,” even though by convention the techniques about to be described are called *wavelet transforms*, to follow the analogy to Fourier analysis. We will remain consistent with the Fourier analysis terminology, but will also continue to point out conventional names found in the wavelet literature. This method will hopefully allow the reader to more closely follow the analogies, while still allowing comparisons to other discussions presented in the literature. The reader is encouraged to refer to Table 3.1 to avoid confusion.

a. Continuous-Time Representation

This representation of the wavelet series, called the *discrete wavelet transform*, will be developed by analogy with its counterpart in Fourier analysis, the Fourier Series. Consider the Fourier series equations pair reproduced here for convenience:

Common name	Consistent name	Time, C or D	Transform, C or D	Input periodic	Output periodic
FS	CTFS	C	D	Yes	No
DFT	DTFS	D	D	Yes	Yes
DTFT	DTFT	D	C	No	Yes
FT	CTFT	C	C	No	No
DWT	CTWS	C	D	Yes or No	Yes or No
DTWT	DTWS	D	D	Yes or No	Yes or No
-	DTWT	D	C	No	No
CWT	CTWT	C	C	No	No

Table 3.1: (C), Continuous, and (D), discrete, periodic and non-periodic input and output relations for the Fourier and wavelet transforms. After Ref. [11]

$$x(t) = \sum_{k=-\infty}^{+\infty} a_k e^{jk2\pi f_0 t} \quad (3.14)$$

$$a_k = \frac{1}{T_0} \int_{T_0} x(t) e^{-jk2\pi f_0 t} dt \quad (3.15)$$

In the case of the Fourier series analysis equation, we have a set of basis functions, $e^{-jk2\pi f_0 t}$, where $k = 0, \pm 1, \pm 2, \dots$. In wavelet analysis, we have a set of basis functions derived from our *wavelet function*, $\psi(t)$, which is also referred to as the generating wavelet, mother wavelet, or analyzing wavelet [11]. As discussed at the beginning of section III-B, wavelets are useful for transient, as well as, dynamic sinusoidal signal analysis. This suggests that a wavelet transform will be a function of two parameters, namely frequency and time, much like the STFT. The *first* parameter introduced will position the wavelet in time by integer translations as shown below:

$$\psi_\tau(t) = \psi(t - \tau), \quad \tau = 0, \pm 1, \pm 2, \dots \quad (3.16)$$

The remaining parameter to be considered is frequency. Multiresolution analysis techniques allow the window size, Δt , and the bandwidth, Δf , to vary across the time-frequency plane within the confines of the uncertainty principle. Variations are obtained in wavelet analysis by forcing the ratio of the bandwidth, Δf , to center frequency, f_c , to be constant. This constraint results in “constant-Q” filtering and the time-frequency plane partitioning shown in Figure 3.5 [7]. Figure 3.5 also illustrates the good time resolution, but poor frequency resolution obtained at high frequencies. Conversely, low frequencies display good frequency resolution, but poor time resolution.

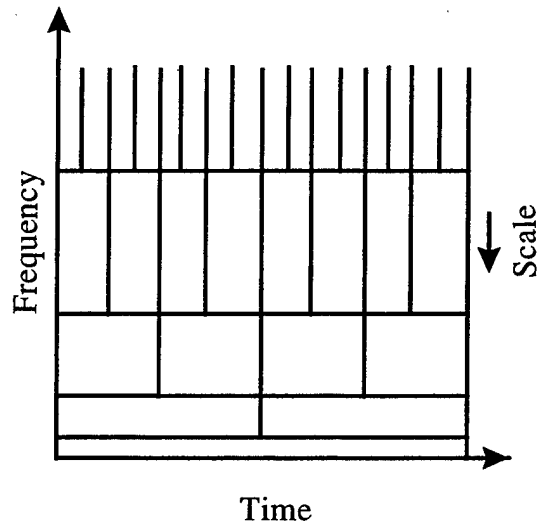


Figure 3.5: The tiling of the time-frequency plane by the wavelet transform.

Thus, the time-frequency tiling described above is implemented by scaling the mother wavelet by a scaling factor, a , where a is inversely proportional to frequency, f . Our *wavelet function* can now be written as:

$$\psi_{a,\tau}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-\tau}{a}\right), \quad (3.17)$$

where the $1/\sqrt{a}$ term is used for energy normalization [7]. The relationships between scale, a , frequency, f , time resolution and frequency resolution are summarized in Table 3.2. Now, restricting a to dyadic scales only (i.e., $a = 2^i$) and let $\tau = n2^i$, we can write our

SCALE	FREQUENCY	FREQUENCY RESOLUTION	TIME RESOLUTION
low	high	poor	good
high	low	good	poor

Table 3.2: Relationships between scale, frequency and resolution.

wavelet series expansion of a given signal, $x(t)$, as:

$$x(t) = \sum_{i,n} c_{i,n} \frac{1}{\sqrt{2^i}} \psi^* \left(\frac{t}{2^i} - n \right). \quad (3.18)$$

The two-dimensional set of coefficients, $c_{i,n}$, are called the *discrete wavelet transform* of $x(t)$, and are analogous to the Fourier series coefficients, a_n , in Equation 3.4.

We mentioned earlier that the *wavelet function* is also sometimes referred to as the "mother" wavelet. This term hints to the existence of another type of wavelet, the "father" wavelet. This function is more commonly referred to as the *scaling function*. We define the scaling function, $\phi(t)$, much as we did the *wavelet function* in Equation 3.24, namely:

$$\phi_{a,\tau}(t) = \frac{1}{\sqrt{a}} \phi \left(\frac{t-\tau}{a} \right), \quad (3.19)$$

or, again using $a = 2^i$ and $\tau = n2^i$ we have:

$$\phi_{i,n}(t) = \frac{1}{\sqrt{2^i}} \phi \left(\frac{t}{2^i} - n \right), \quad (3.20)$$

Define the subspace of L^2 spanned by these scaling functions as:

$$V_i \equiv \text{Span}_n \left\{ \phi_n \left(\frac{1}{2^i} t \right) \right\}. \quad (3.21)$$

Similarly, the subspace of L^2 spanned by the wavelet functions can be defined as:

$$W_i \equiv \text{Span}_n \left\{ \psi_n \left(\frac{1}{2^i} t \right) \right\}, \quad (3.22)$$

where W_i is the orthogonal complement of V_i in subspace V_{i-1} . Therefore, the relationships of these subspaces is given by:

$$L^2 = V_N \oplus W_N \oplus \dots \oplus W_2 \oplus W_1, \quad (3.23)$$

where V_N is the final space spanned by the scaling function which is usually chosen to represent the coarsest detail of interest. The relationships between these subspaces are illustrated in Figures 3.6 and 3.8. Equation 3.23 allows us to rewrite Equation 3.18 in terms of both the wavelet and scaling functions for any function $x(t) \in L^2$ as:

$$x(t) = \sum_{n=-\infty}^{+\infty} a(n)\varphi_n(t) + \sum_{i=1}^N \sum_{n=-\infty}^{+\infty} d(i,n)\psi_{i,n}(t). \quad (3.24)$$

Note here that $a(n)$ are the coefficients of the scaling function, $\varphi_n(t)$, which spans V_N of Equation 3.23. Similarly, $d(i,n)$ represent the N sets of coefficients of the wavelet functions, $\psi_{i,n}(t)$, which span N subspaces, W_i , where $i=1,2,\dots,N$.

The coefficients $a(n)$ and $d(i,n)$ define the *discrete wavelet transform* of $x(t)$ in terms of both the scaling and wavelet functions, just as the coefficients, $c_{i,n}$, of

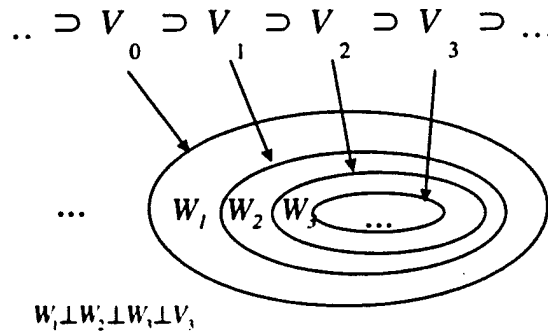


Figure 3.6: Orthogonal Vector Spaces V_i and W_i representing the spaces spanned by the scaling function $\varphi_{i,n}(t)$ and $\psi_{i,n}(t)$, respectively. After Ref. [11]

Equation 3.18 did in terms of just the wavelet function. If these basis functions $\varphi(t)$ and $\psi(t)$ form an orthonormal basis or at least a tight frame [11], the coefficients may be derived as:

$$a(n) = \int x(t) \varphi_n(t) dt, \quad (3.25)$$

and:

$$d_i(n) = \int x(t) \psi_{i,n}(t) dt. \quad (3.26)$$

b. Discrete-Time Representation

The nesting of subspaces, as shown in Figure 3.6, is achieved with the *dilation equation* given by:

$$\varphi(t) = \sum_n h(n) \sqrt{2} \varphi(2t - n), \quad n = 0, \pm 1, \pm 2, \dots \quad (3.27)$$

where $h(n)$ is the scaling function coefficients and $\sqrt{2}$ maintains the norm of the scaling function as each successive dyadic scale is calculated [11]. Similarly, we can define the *wavelet function* in terms of the *scaling function* by:

$$\psi(t) = \sum_n h_1(n) \sqrt{2} \varphi(2t - n), \quad n = 0, \pm 1, \pm 2, \dots \quad (3.28)$$

where,

$$h_1(n) = (-1)^n h(N - 1 - n), \quad (3.29)$$

where N is the finite even length of $h(n)$, as shown in Burrus [11]. Equation 3.27 can be interpreted as obtaining lower resolution by down-sampling by two after lowpass filtering with the half-band filter with impulse response $h(t)$. Similarly, Equation 3.28 can be

interpreted as obtaining lower resolution by down-sampling by two after highpass filtering with the half-band filter with impulse response $h_I(t)$. The analysis filter bank just described are illustrated in Figure 3.7.

The coefficients of Equations 3.27 and 3.28, namely $h(n)$ and $h_I(n)$, can be viewed as digital filters and the coefficients of Equation 3.24, namely $a(n)$ and $d_I(n)$, can be viewed as digital signals. With this view it can be shown [11] that the coefficients of Equations 3.25 and 3.26 become:

$$a_{i+1}(n) = \sum_m h(m-2n)a_i(m), \quad (3.30)$$

and:

$$d_{i+1}(n) = \sum_m h_I(m-2n)a_i(m), \quad (3.31)$$

respectively. The above discussion leads to what is commonly referred to as the filter bank view of the DWT or the *discrete time wavelet transform* (DTWT) [11]. These last two equations show that we obtain the coefficients at the next level of scale, $i+1$, by convolving the approximation coefficients, $a_i(m)$, at level i with the time reversed filter coefficients, $h(-n)$ and $h_I(-n)$, and then down-sampling by two. This idea is illustrated in block diagram form in Figure 3.7, where $\downarrow 2$ indicates decimation, or down-sampling, by two and the other blocks represent filtering operations. As mentioned previously, $h(n)$ is a lowpass filter and $h_I(n)$ is a highpass filter. Originally, we have a signal, $x[n]$. For this case, if the samples of the input function, $x[n]$, are above the Nyquist rate, they represent a good approximation to the scaling coefficients at that scale, meaning that the

coefficients, a_0 , equal $x[n]$ [11]. Next, we filter a_0 with $h(n)$ and $h_I(n)$ and down-sample by two, to get the next scale level coefficients, a_1 and d_1 , respectively. The resulting coefficients represent the next coarser scale level which give better frequency resolution, but poorer time resolution. We can repeat the process with the approximation coefficients, a_1 , to get a_2 and d_2 , as indicated in Figure 3.7, which represent the next coarser scale level with even better frequency resolution at the price of even poorer time resolution. Normally, this process is repeated until the desired level of coarseness, or frequency resolution, is obtained.

Wavelet transforms can also be described in terms of filter banks as shown in Figure 3.7. Let $i=0$ for the example to follow. When the sampled signal is passed through the first filter bank consisting of $h_I(-k)$ and $h(-k)$, the highpass filter and lowpass filter, respectively, the original space is divided into W_1 and V_1 as shown in Figure 3.8a. After passing through the second stage of the filter bank, V_1 is divided into W_2 and V_2 . If we assume that we have now reached our coarsest scale desired, we now have the entire signal space of interest represented by $W_1 \oplus W_2 \oplus V_2$, as shown in Figure 3.8b. If we have

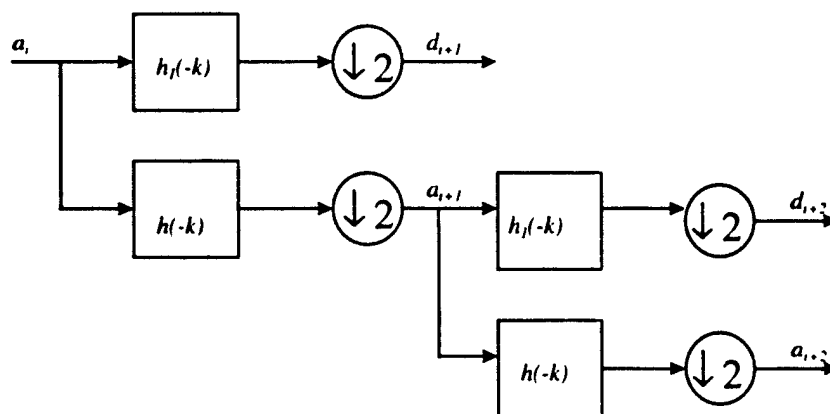


Figure 3.7: Analysis filter bank for calculating the discrete time wavelet transform.

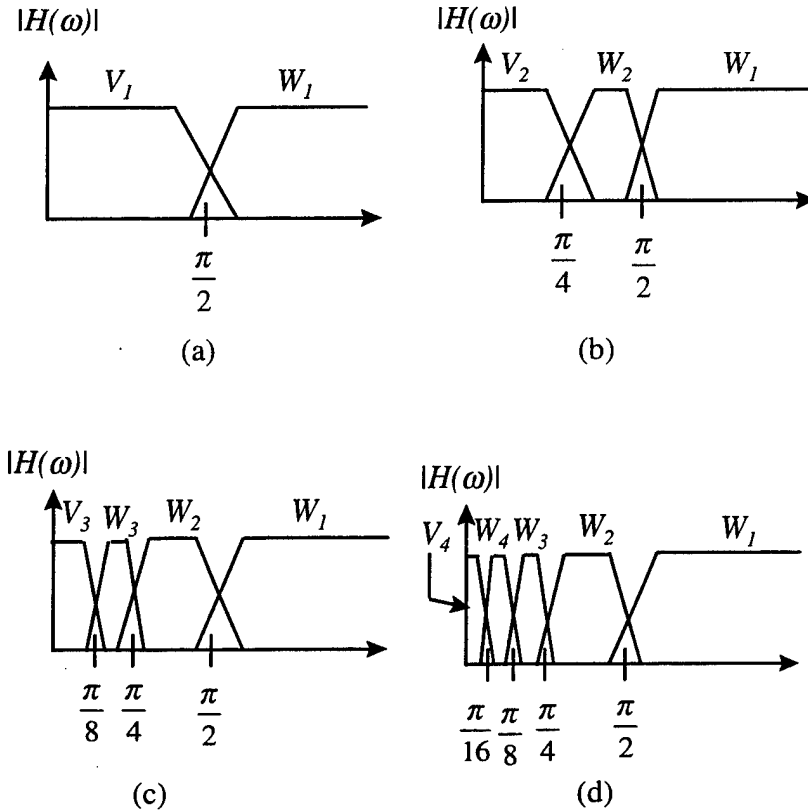


Figure 3.8: Spectral Partitioning by analysis filter bank.

not reached the coarsest scale desired, we continue to iterate the filter bank to further partition the frequency bands, as shown in Figures 3.8c and 3.8d. Note that the gain of the frequency response in Figure 3.8 has been normalized for all bands. In actuality, the areas under the curves would be equal in order to meet the requirements for conservation of energy.

Synthesis is performed by simply reversing the process as shown in the filter bank representation of Figure 3.9 and given by:

$$a_i(n) = \sum_m a_{i+1}(m)h(n-2m) + \sum_m d_{i+1}(m)h_1(n-2m). \quad (3.32)$$

Note in Figure 3.9 that $\uparrow 2$ indicates up-sampling by two by inserting a zero between each point, and that the result is filtered with $h(n)$ and $h_l(n)$ not their time reversed counterparts. The analysis Equations 3.30 and 3.31 taken together with the synthesis Equation 3.32 are known as Mallat's algorithm [15]. Recall that the two transforms we have just introduced, called the DWT and the DTWT, are in the Fourier analysis terminology called the *continuous time wavelet series* and the *discrete time wavelet series*, respectively. Therefore, we are left to draw an analogy with the “continuous” Fourier transforms (see Table 3.1) .

2. Wavelet Transform

a. Continuous-Time Representation

Recall that the STFT of Equation 3.12 is of the form:

$$STFT(\tau, f) = \int_{-\infty}^{+\infty} x(t) \psi^*(t - \tau) dt, \quad (3.33)$$

where $\psi(t) = g(t)e^{-2j\pi_0 t}$. If we introduce the scaling factor a into Equation 3.33, we obtain the *continuous time wavelet transform* (CTWT) expansion given by:

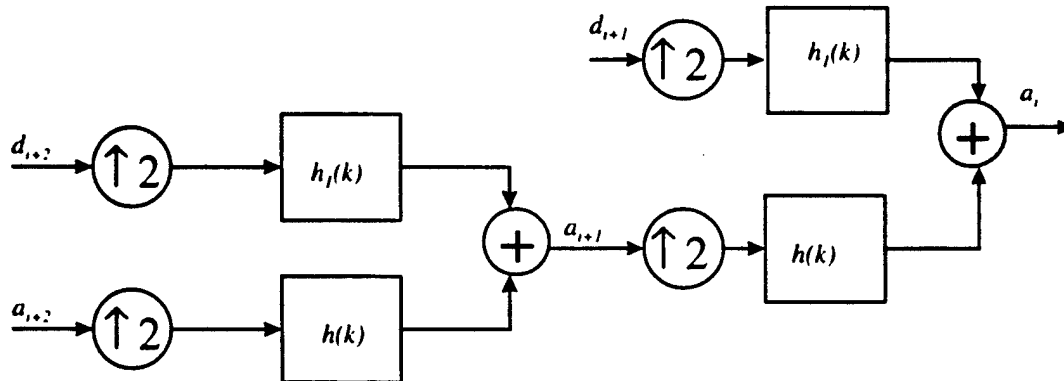


Figure 3.9: Synthesis filter bank for calculating the discrete time wavelet transform.

$$\text{CWT}(\tau, a) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} x(t) \psi^* \left(\frac{t - \tau}{a} \right) dt, \quad (3.34)$$

where $a = f_0/f$ and the term $1/\sqrt{a}$ term is introduced for energy normalization [7].

Although not used in this work, one can define the inverse CWT as:

$$x(t) = K \iint \frac{1}{a^2} \text{CWT}(a, \tau) \psi \left(\frac{t - \tau}{a} \right) da d\tau, \quad (3.35)$$

where the normalizing constant is:

$$K = \int \frac{|\Psi(f)|^2}{|2\pi f|} df, \quad (3.36)$$

and $\Psi(f)$ is the Fourier transform of $\psi(t)$ [11].

b. Discrete-Time Representation

Signal processing in most real world applications is performed on a digital computer. This implies that we must use the discrete, sampled, version, $x[n]$, of the continuous signal, $x(t)$, which we wish to process. This necessity leads us to the need for a *discrete time continuous wavelet transform* (DTCWT), or to be consistent with Fourier analysis terminology, simply a *discrete time wavelet transform* (see Table 3.1). The actual analysis and synthesis Equations for the DTCWT will not be presented here, but rather the important conceptual differences with the DWT will be discussed.

Recall that for the DWT that the scales, a , are restricted to *dyadic* scales (i.e., $a = 2^i$, for $i = 0, \pm 1, \pm 2, \dots$) to allow for the filter bank implementation. However, no such restriction needs to be made for the DTCWT. The scale ranges, continuously,

from that of the original signal, up to whatever maximum the user desires [16]. This continuous range of scale is the first important difference compared to the DWT. The second significant difference is that shifting of the wavelet function by the DTCWT is continuous [16]. The analyzing wavelet is shifted just as smoothly over the time domain of the signal being analyzed as it is scaled over the frequency domain. In the next chapter we will look at the abilities and limitations of the DTCWT in detecting transients by using the *Wavelet Toolbox*'s implementation of this transform.

So far we have introduced frequency hopping signals and the various forms of *wavelet analysis*. In the next chapter we will apply the latter to the former to see how wavelet analysis might be used in the detection and estimation of frequency hopping signals.

IV. WAVELET BASED TRANSIENT DETECTION

The frequency hopping detection scheme considered in this work involves the detection of discontinuities. This chapter, first, briefly reviews how wavelets can be used to detect signal discontinuities. Next, it defines the temporal correlation function used in our detection scheme. Finally, it introduces several preprocessing tools used to minimize potential performance degradation when additive white Gaussian noise is present in the communication signal.

A. DETECTING DISCONTINUITIES USING WAVELETS

Consider the three signals given in Figure 4.1. Figure 4.1a represents a signal, $x(t)$, with a discontinuity, Figure 4.1b represents a signal with a discontinuity in the first derivative, and Figure 4.1c represents a signal with a discontinuity in the second derivative. Wavelets may be used to detect each of these discontinuities, if the chosen wavelet is able to represent the highest order derivative present in the signal function, as any wavelet with, at least, p vanishing moments can be used to detect a discontinuity in the $p-1$ derivative [10]. However, the wavelet's ability to detect these discontinuities erodes quickly in the presence of noise, as will be shown later.

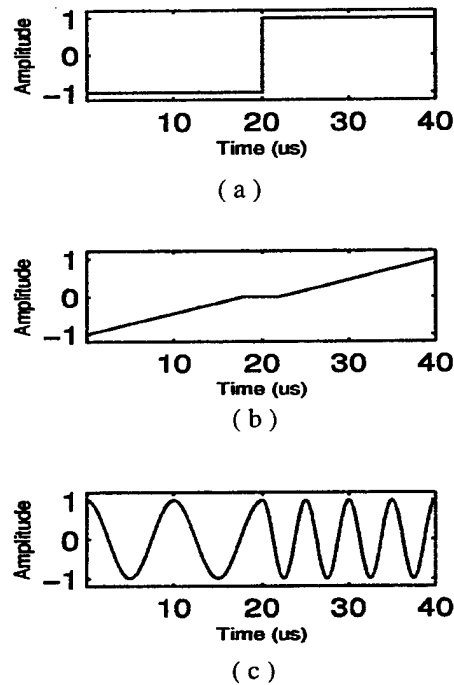


Figure 4.1: Function discontinuities (a) in the function (step function), (b) in the first derivative (interrupted ramp function), and (c) in the second derivative (frequency hopping function).

Now, consider the various wavelets shown in Figure 4.2. The Morlet wavelet is an example of a continuous wavelet that can represent all three of the functions in Figure 4.1, if appropriately compressed or dilated (i.e., it can be compressed to until it contains a vertical portion, dilated to match a ramp region, or dilated less to match the sinusoidal terms). The Haar wavelet shown in the figure has only one, $p=1$, vanishing moment and, therefore, can only detect discontinuities in the zero, $p-1=0$, derivative (i.e. the function itself). Further, note the Haar wavelet does contain a portion which can match functions with discontinuities, such as the step function of Figure 4.1, exactly, but no portions which can match the ramp or sinusoidal functions of Figure 4.1. Therefore, it is not able to detect discontinuities in either the first or second derivatives, as shown in the next

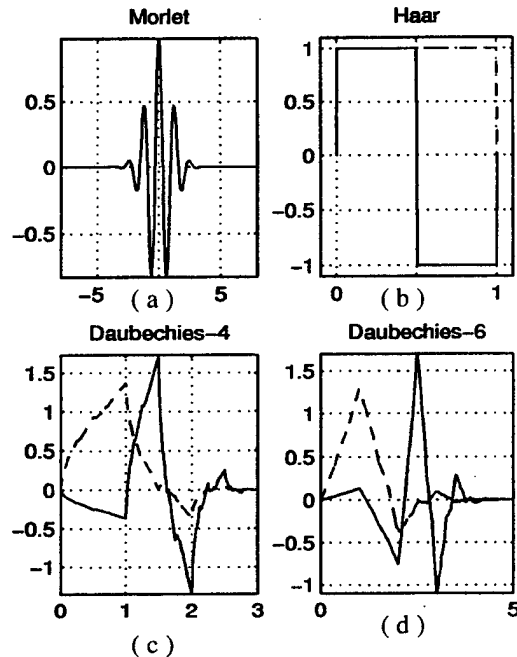


Figure 4.2: The (a) Morlet, (b) Haar, (c) Daubechies-4 and (d) Daubechies-6 wavelets with their scaling functions shown as dashed lines.

section. Similar arguments hold for the Daubechies-4 ($p=2$) and Daubechies-6 ($p=3$) wavelets shown in Figure 4.2 which are able to detect discontinuities in functions or their derivatives up to the first and second degree, respectively.

1. The Continuous Wavelet Transform

As mentioned earlier, the *continuous wavelet transform* (CWT) can be used to detect discontinuities in a function and its first and second derivatives. The *discrete time continuous wavelet transform* of Subsection III.B.2.b was used to detect the discontinuities of the functions shown in Figure 4.1, using the Morlet wavelet. Figure 4.3. shows that it is able to detect discontinuities in the function shown in Figure 4.1a, in the first derivative of the function shown in Figure 4.1b, and in the second derivative of

the function shown in 4.1c. However, since the CWT is computationally expensive, we will choose not to use it in our detection algorithm.

2. The Discrete Wavelet Transform

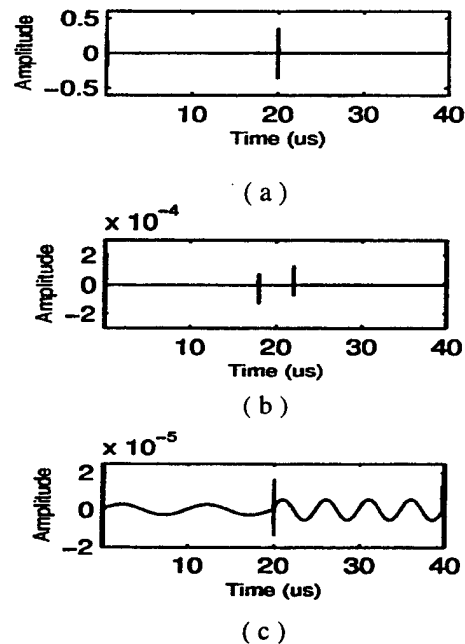


Figure 4.3: The first level detail coefficients, d_1 , from the Continuous Wavelet Transform of the functions of Figure 4.1, (a) the step function, (b) the interrupted ramp function, and (c) the frequency hopping function, using the Morlet wavelet of Figure 4.2a.

Recall that the Daubechies-6 wavelet has three vanishing moments allowing it to detect discontinuities in a function and discontinuities in both the first and second derivative of a function. Figure 4.4 shows that the Daubechies-6 was indeed able to detect such discontinuities in the functions shown in Figure 4.1 using the DTWT of section III.B.1.e. Further, recall that the Daubechies-4 and the Haar wavelets have only two and one vanishing moments, respectively. As shown in Figure 4.5, the Daubechies-4

wavelet was only able to detect the discontinuities in the step and interrupted ramp functions, but not the discontinuity in the second derivative of the frequency hopping function. Figure 4.6 shows the results of performing a DWT on the functions shown in Figure 4.1 using a Haar wavelet. As expected, it was only able to detect the discontinuity in the step function.

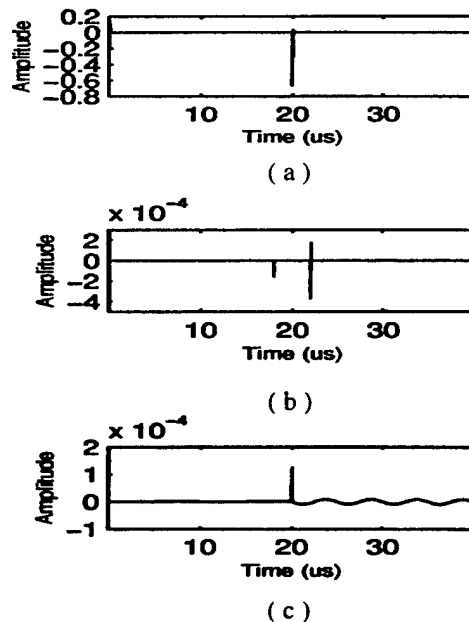


Figure 4.4: The first level detail coefficients, d_1 , from the Discrete Wavelet Transform of the functions from Figure 4.1, (a) the step function, (b) the interrupted ramp function, and (c) the frequency hopping function, using the Daubechies-6 wavelet of Figure 4.2d.

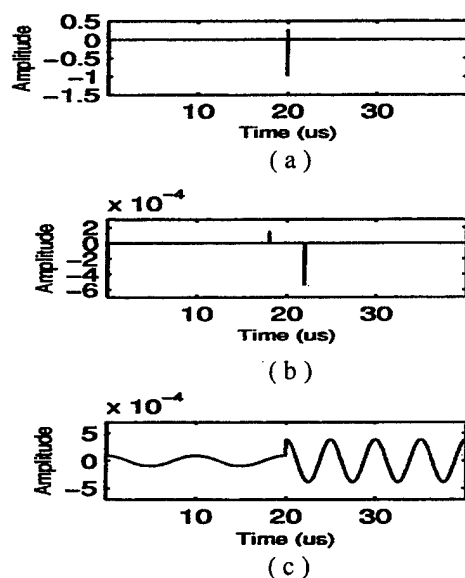


Figure 4.5: The first level detail coefficients, d_1 , from the Discrete Wavelet Transform of the functions of Figure 4.1, (a) the step function, (b) the interrupted ramp function, and (c) the frequency hopping function, using the Daubechies-4 wavelet of Figure 4.2c.

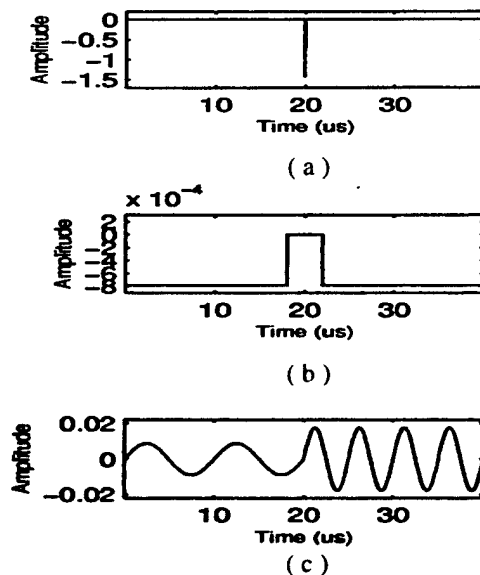


Figure 4.6: The first level detail coefficients from the Discrete Wavelet Transform of the functions in Figure 4.1, (a) the step function, (b) the interrupted ramp function, and (c) the frequency hopping function using the Haar wavelet of Figure 4.2b.

3. The Continuous Wavelet Transform Applied to Functions with Additive Noise

Adding white Gaussian noise (AWGN) to the signal, $x(t)$, to obtain a SNR of 60 dB, results in the signal shown in Figure 4.7. Note that the noise is not even perceptible

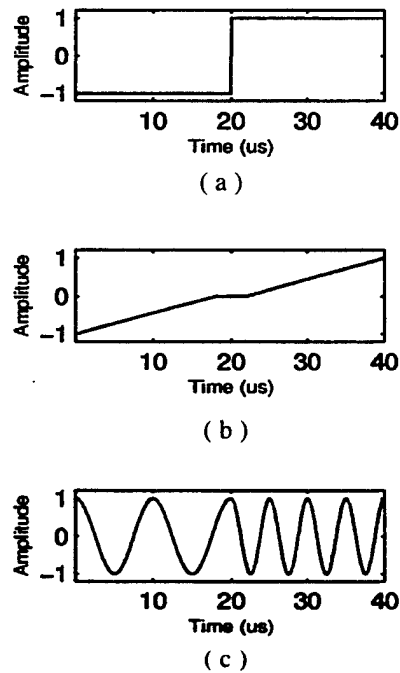


Figure 4.7: Functions of Figure 4.1 embedded in 60 dB of AWGN.

with the given figure's resolution. However, by examining the wavelet transforms of these functions with this moderate noise, insight may be gained into the robustness of these discontinuity detection techniques. Figure 4.8 shows the results obtained when using the DTCWT with the Morlet wavelet. Note, that even though it is still able to

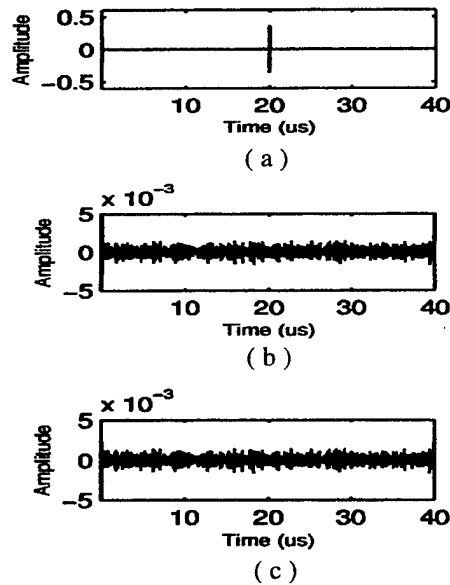


Figure 4.8: The first level detail coefficients, d_1 , from the DTCWT on the functions of Figure 4.7 using the Morlet wavelet.

detect a discontinuity in the step function in noise, it fails to detect discontinuities caused by the first and second derivatives of the functions of Figures 4.7b and 4.7c, respectively.

4. The Discrete Wavelet Transform Applied to Functions with AWGN

Applying the DTWT using Daubechies-6, Daubechies-4, and Haar wavelets to the functions shown in Figure 4.7 leads to the results shown in Figure 4.9 to 4.11. We note that discontinuities in the first and second derivatives are not detected due to noise degradations present in the signals. Therefore, one can conclude that a scheme based on detecting discontinuities in the function itself, vice its derivatives, is more robust.

Thus, the next question becomes: when all these wavelets can detect these discontinuities in the function itself, how does one decide which wavelet to use when looking for such discontinuities? The DTWT computed using filter coefficients, $h(n)$ and $h_1(n)$, of short length are not only less expensive computationally, but also produce better

time localization [10]. Based on the latter consideration the Haar wavelet should be a good candidate, since it has the shortest filters of the three.

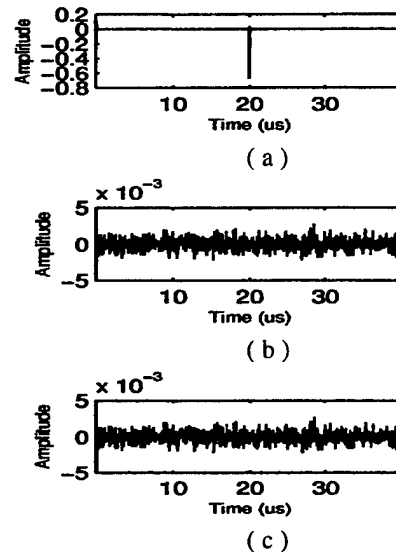
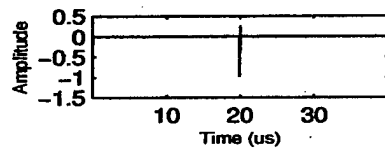


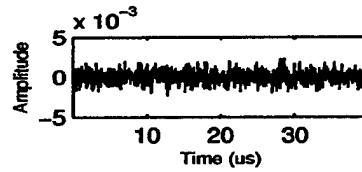
Figure 4.9: The first level detail coefficients, d_1 , from the DTWT of the functions of Figure 4.7 using the Daubechies-6 wavelet.

5. Averaging of Scales

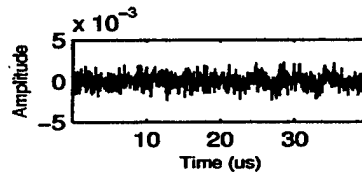
The averaging of several scales can enhance a wavelet's ability to detect discontinuities in noise. The idea is that for true discontinuities, the spikes will line up across all scales, while the spikes due to noise will not line up. Consider, for example, the stair step function shown in Figure 4.12a and the same function in noise in Figure 4.12b. The Haar wavelet detects the discontinuities perfectly in the no-noise case, as shown in Figure 4.12c. However, the performance is seriously degraded by the noise, as shown in Figure 4.12d.



(a)

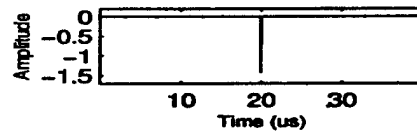


(b)

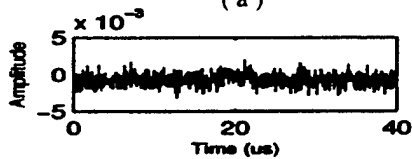


(c)

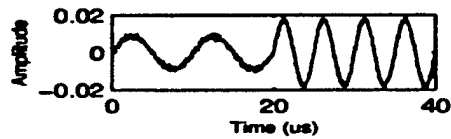
Figure 4.10: The first level detail coefficients, d_1 , from the DTWT of the functions of Figure 4.7 using the Daubechies-4 wavelet.



(a)



(b)



(c)

Figure 4.11: The first level detail coefficients, d_1 , of the DTWT of the functions of Figure 4.7 using the Haar wavelet.

Figure 4.13 shows the DTWT of the stair step function in noise at scales 1 through 6 computed using the Haar wavelet. Note that all step times can be detected if the wavelet coefficients obtained from the first six scales are summed and clipped at some threshold, as shown in the top middle plot of Figure 4.13 labeled "Output of detector" [10].

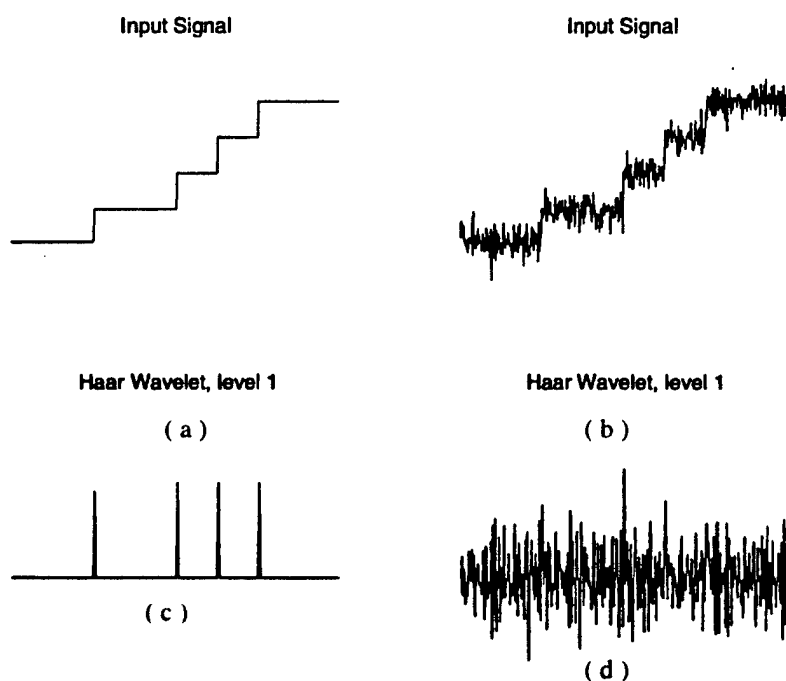


Figure 4.12: Detection of the stair step function (a) with no noise, (b) in noise with the corresponding detector output shown in (c) and (d), respectively. From Ref. [10].

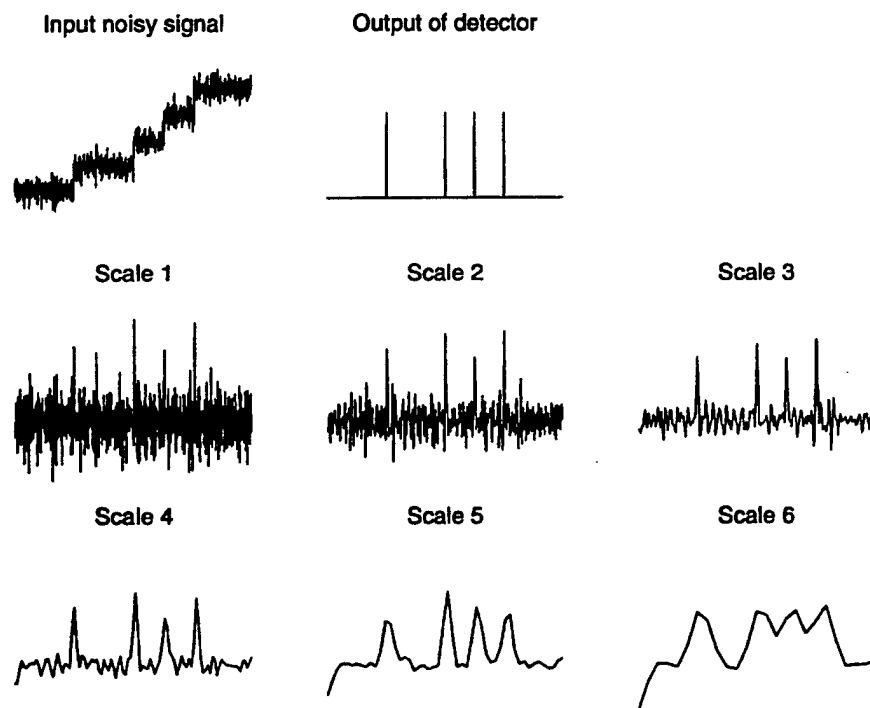


Figure 4.13: Detection of stair step function in noise using scale averaging. From Ref. [10].

B. THE TEMPORAL CORRELATION FUNCTION

The *temporal correlation function* (TCF) of a signal $x(t)$ is defined as:

$$\text{TCF}_x(\mathbf{t}, \tau) = x\left(\mathbf{t} + \frac{\tau}{2}\right) x^*\left(\mathbf{t} - \frac{\tau}{2}\right), \quad (4.1)$$

where \mathbf{t} is the absolute center time and τ is the lag time. This function has been shown useful when analyzing nonstationary signals [17]. Note that the TCF is conjugate symmetric along the τ axis as:

$$\text{TCF}(\mathbf{t}, \tau) = \text{TCF}^*(\mathbf{t}, -\tau), \quad (4.2)$$

where $*$ represents the complex conjugate [17]. Therefore, no additional information is gained by calculating the function for negative values of the lag, τ . In addition, note that the TCF of a real signal may exhibit interference terms due to its nonlinear definition. For example, consider a real sinusoidal signal $x(t)$:

$$x(t) = \sin(2\pi f t + \theta). \quad (4.3)$$

The resulting TCF of $x(t)$ is given by:

$$\text{TCF}(\mathbf{t}, \tau) = \frac{1}{2} \left[\cos(2\pi f \mathbf{t}) - \cos(2\pi [2f] \mathbf{t} + 2\theta) \right]. \quad (4.4)$$

The first term inside the square brackets consists of auto-terms at the frequency, f , of $x(t)$, while the second term consists of interference terms at twice the frequency, f , of $x(t)$ [17].

Note, however, that the TCF of the *analytic* signal, $x_a(t)$, obtained from $x(t)$ given by:

$$x_a(t) = e^{j2\pi f_1 t + \theta}, \quad (4.5)$$

leads to the TCF function:

$$\text{TCF}_{x_a}(t, \tau) = e^{j2\pi f_1 \tau}. \quad (4.6)$$

Equation 4.6 shows that the TCF expression contains only auto terms at the frequency of $x_a(t)$ [Fried]. For this reason the analytic form of the signal, $x(t)$, is often preferable to its real counterpart and is the form we will choose to work with in our detection scheme to be developed.

Next, consider the nonstationary analytic signal given by:

$$x_a(t) = e^{j2\pi f_1 t} [u(t) - u(t - T_{hop})] + e^{j2\pi f_2 t} [u(t - T_{hop} + 1) - u(t - T)], \quad (4.7)$$

for $0 \leq t \leq T$, where T_{hop} is the time of the hop (or change in frequency) from f_1 to f_2 , and where $u(t)$ is the unit step function given by:

$$u(t) = \begin{cases} 1 & , \text{for } t \geq 0 \\ 0 & , \text{for } t < 0 \end{cases} \quad (4.8)$$

Substituting Equation 4.7 into Equation 4.1 leads to:

$$\begin{aligned}
TCF(t, \tau) = & \left\{ \begin{aligned} & e^{j2\pi f_1 \tau} \left[\begin{aligned} & u\left(t + \frac{\tau}{2}\right) \left[u\left(t - \frac{\tau}{2}\right) - u\left(t - \frac{\tau}{2} - T_{hop}\right) \right] \\ & + u\left(t + \frac{\tau}{2} - T_{hop}\right) \left[u\left(t - \frac{\tau}{2} - T_{hop}\right) - u\left(t - \frac{\tau}{2}\right) \right] \end{aligned} \right] \\ & + e^{j2\pi f_2 \tau} \left[\begin{aligned} & u\left(t + \frac{\tau}{2} - T_{hop} + 1\right) \left[u\left(t - \frac{\tau}{2} - T_{hop} + 1\right) - u\left(t - \frac{\tau}{2} - T\right) \right] \\ & + u\left(t + \frac{\tau}{2} - T\right) \left[u\left(t - \frac{\tau}{2} - T\right) - u\left(t - \frac{\tau}{2} - T_{hop} + 1\right) \right] \end{aligned} \right] \\ & + e^{j2\pi \left[(f_2 - f_1)t + \left(\frac{f_1 + f_2}{2}\right)\tau \right]} \left[\begin{aligned} & u\left(t + \frac{\tau}{2} - T_{hop} + 1\right) \left[u\left(t - \frac{\tau}{2}\right) - u\left(t - \frac{\tau}{2} - T_{hop}\right) \right] \\ & + u\left(t + \frac{\tau}{2} - T\right) \left[u\left(t - \frac{\tau}{2} - T_{hop}\right) - u\left(t - \frac{\tau}{2}\right) \right] \end{aligned} \right] \end{aligned} \right\}, \quad (4.9) \\
= & TCF_1(t, \tau) + TCF_2(t, \tau) + TCF_{12}(t, \tau),
\end{aligned}$$

where $TCF_1(t, \tau)$, $TCF_2(t, \tau)$, and $TCF_{12}(t, \tau)$ represent the 1st, 2nd, and 3rd terms of Equation 4.9. Figure 4.14a presents a phase plot of Equation 4.9 for some arbitrary f_1 and f_2 . The combinations of the different shifted versions of the unit step functions, $u(t)$, force the TCF to only take on values within the regions shown in Figures 4.14b to 4.14d. Note that $TCF_1(t, \tau)$ is a function of f_1 and τ only. The second term, $TCF_2(t, \tau)$, is a function of f_2 and τ only, while the last term, $TCF_{12}(t, \tau)$, is a function of f_1, f_2, t , and τ . Note that the frequency hopping time, $T_{hop} = 100$, is located where the region covered by $TCF_1(t, \tau)$ ends and the region covered by $TCF_2(t, \tau)$ begins. Further, note that for a given value of τ , the terms within the triangles (i.e., the regions where $TCF_1(t, \tau)$ and $TCF_2(t, \tau)$ are defined) are constant, although at different levels. This fact is further illustrated in Figure 4.15a for a few values of τ , and will be exploited in our detection algorithm. Equation 4.9 showed that the phase behavior within the cross-terms region, $TCF_{12}(t, \tau)$ is linear.

The phase expression is periodic over 2π producing discontinuities at regular intervals across $\text{TCF}_{12}(\mathbf{t}, \tau)$. It is important to realize, however, that the period of these intervals is a function of f_1 and f_2 and, therefore, not predictable without knowing f_1 and f_2 , which in general we do not. Nevertheless, for any given value of the lag, $\tau < T_{hop}$, this region of cross-terms is centered on the hop time, T_{hop} , another fact which may be exploited.

It should be further highlighted that within a given auto-term triangle, there is only

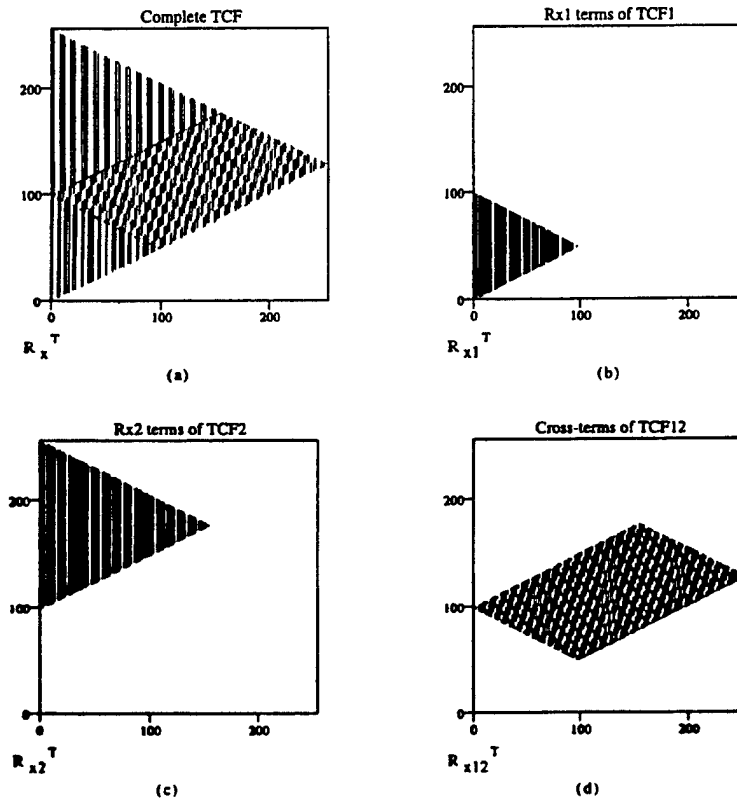


Figure 4.14: Phase plot of the Temporal Correlation Function obtained from $x_s(t)$ defined in Equation 4.7. (b) shows the portion of (a) which contains the auto-terms, TCF_1 , due to f_1 only, (c) shows the portion of (a) which contains the auto-terms, TCF_2 , due to f_2 only, and (d) shows the portion of (a) which contains the cross-terms, TCF_{12} , due to both f_1 and f_2 .

one frequency component along the τ axis. Therefore, if one were to calculate the frequency response **within** one of these triangles, across the values of τ , an estimate of the frequency during the period of t , covered by the given triangle could be extracted.

So far, only signals without the presence of noise have been considered. In practice, some amount of noise is always present and the following section introduces additional preprocessing tools designed to facilitate the extraction of frequency hopping features in noisy environments.

C. PREPROCESSING TECHNIQUES

As mentioned before, noise added to the signal impedes our ability to detect discontinuities in frequency hopping signals, or in general, to detect transients. This section investigates how the additive noise alters the TCF expression, and presents

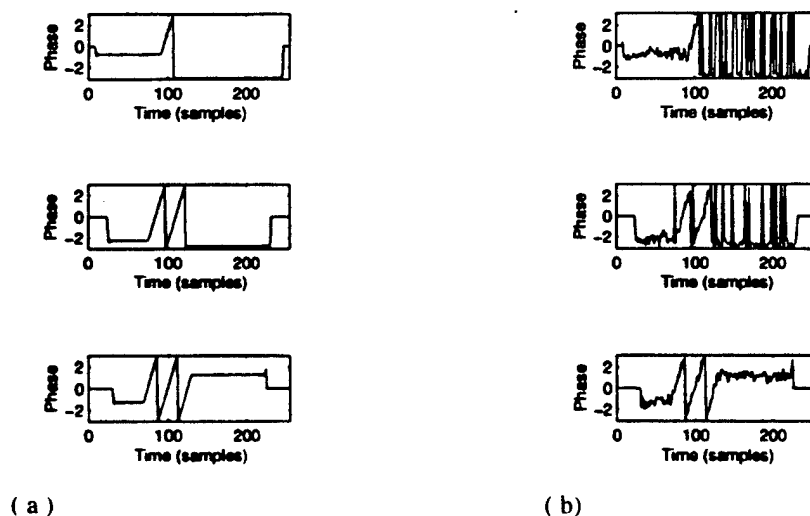


Figure 4.15: Phase behavior of the TCF at lags 16 (top), 48 (middle), and 60 (bottom) for (a) the case with no noise and (b) the case with additive WGN.

several preprocessing techniques to minimize the noise effects. Specifically, we will investigate how the noise affects the phase information in the TCF computed on the analytic signal, $x_a(t)$, given in Equation 4.7. Recall that the frequency value information and the hopping time may be obtained using the phase of the TCF computed from an analytical frequency hopping signal, $x(t)$, as shown earlier. In addition, note that:

- 1) the phase components of the auto-terms, $\text{TCF}_1(\mathbf{t}, \tau)$ and $\text{TCF}_2(\mathbf{t}, \tau)$, have constant values for a given lag time τ ,
- 2) the phase components of the cross-terms, $\text{TCF}_{12}(\mathbf{t}, \tau)$, are linear in \mathbf{t} for a given lag time τ .

Figure 4.15 plots the phase information obtained from the TCF expression shown in Figure 4.14 for lags $\tau = 16, 48$ and 60 in noise free and noisy environments. Note that the hopping time ($T_{hop} = 100$) occurs in the middle of the linear phase excursion. Also, note that the noise causes random spikes in the phase information. Such spike occurrences must be minimized if we are to produce a robust, reliable detection algorithm.

1. Unwrapping the Phase

The phase, $p(t)$, of a signal, $x(t)$, may be unwrapped as:

$$\text{unwrap}(p(t)) = \begin{cases} p(t) & \text{if } |p(t) - p(t-1)| \leq \pi, \\ p(t) + 2\pi & \text{if } p(t) - p(t-1) < -\pi, \\ p(t) - 2\pi & \text{if } p(t) - p(t-1) > \pi, \end{cases} \quad (4.10)$$

Unwrapping the phase serves two purposes. First, it removes the discontinuities present in the region covered by $\text{TCF}_{12}(t, \tau)$. The resulting unwrapped phase becomes purely linear for a given lag time τ in the region covered by $\text{TCF}_{12}(t, \tau)$, as shown in Figure 4.16a. Second, the unwrap function is, also, very effective at removing the random spikes in the phase due to noise, as shown in Figure 4.16b. However, as we shall see, the wavelet transform is sensitive to even low level noisy behavior. The same properties which make the WT adept at detecting transients make it susceptible to false alarms due to the noise. Therefore, further “noise quieting” techniques prove beneficial.

2. Median Filter

A median filter of size N , is given by:

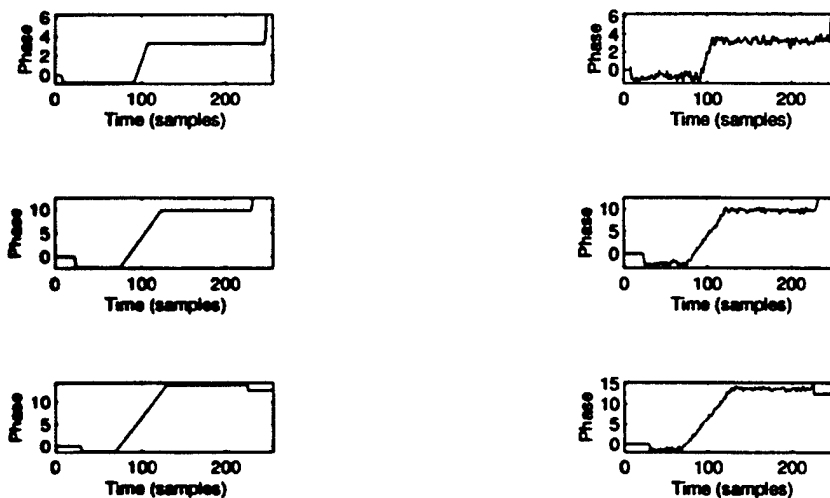


Figure 4.16: Unwrapped phase behavior at lags τ equal to 16 (top), 48 (middle), and 60 (bottom) for (a) the case with no noise and (b) the case with additive WGN (SNR = 10 dB).

$$x_{MF}(t) = \text{median}(x(t)w_N(t)), \quad (4.11)$$

where $w_N(t)$ is a window of length N centered on t . The *median filter* is a nonlinear filter with the important characteristic that it does not average and, therefore, preserves discontinuities. Examples of the effects of the median filter on our noisy phase information will be given in the next chapter.

3. Differentiation

Consider a function which is at a constant value, C_1 , for some period of time, $0 \leq t \leq t_1$, then ramps up with slope, m , for another period, $t_1 < t \leq t_2$, and is then assumes another constant value, C_2 , for the remainder of its duration, $t_2 < t \leq T$, as shown in Figure 4.17a for $C_1 = -1.2474$, $C_2 = 13.832$, $t_1 = 70$, $t_2 = 130$, and $T = 256$. For future reference, note that this figure closely resembles that of Figure 4.16a (bottom) with the exception that the end terms, which will be discussed later, have been set to the same constant value as those before and after the ramp (i.e. the constant levels due to the auto-terms of Figure 4.14a). Differentiating the function shown in Figure 4.17a, leads to Figure 4.17b as the constant terms go to zero, and the ramp portion becomes a constant equal to the slope, $m = \frac{C_2 - C_1}{t_2 - t_1} = 0.251$, of the ramp. Therefore we have created a pulse, a function with two discontinuities, of height $m = 0.251$ and width equal to $t_2 - t_1 = 60$ as shown in Figure 4.6b. This is another key feature which we will exploit in our detection algorithm.

The corresponding desired results obtained from the application of these functions are listed as follows:

1. The **TCF** produces a two-dimensional correlation function with the property that when applied to a analytic signal such as the one in Equation 4.7 (i.e. a complex sinusoid with **one** frequency hop) it results in a pattern, as shown in Figure 4.14. Furthermore, the cross-terms, TCF_{12} , are centered on the hopping time, $t=T_{hop}$, for $0 \leq t \leq T$ and $0 \leq \tau \leq T_{hop}$.
2. When the TCF phase is **unwrapped**, for any given lag, τ , it results in a signal which is constant for the period of the first auto-terms, TCF_1 , then is a ramp for the period of the cross-terms, TCF_{12} , and then is constant again, at a different level, for the period of the second auto-terms, TCF_2 .

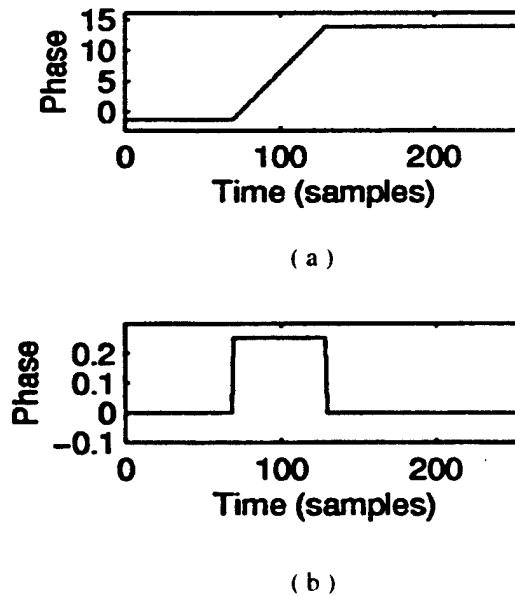


Figure 4.17: A ramp function is shown in (a), and its derivative in (b).

3. The application of a **median filter** to the phase information in noise can smooth out the noise, but with the important characteristic that it will preserve the discontinuities.
4. **Differentiation** of a function of the type described in item two above results in a pulse centered on the hopping time, T_{hop} , for $\tau < T_{hop}$.

Having explored the abilities and limitations of using wavelet transforms to detect discontinuities, and having examined some preprocessing tools, we are now ready to develop our frequency hopping signal detection algorithm.

V. DETECTION AND ESTIMATION ALGORITHM AND SIMULATION

A. DETECTION AND ESTIMATION ALGORITHM

Using the tools described in the last chapter, the algorithm steps for the detection and estimation of frequency hopping signals in noise can now be enumerated. The steps are as follows:

1. Transform the real signal into an analytic signal.
2. Segment data into frames of length less than or equal to the minimum hopping time, T_{hop_min} . This assumption ensures that, at most, one hop will be present in the processing frame.
3. Compute the *temporal correlation function* on each frame.
4. Extract the phase information by calculating the angle of the TCF.
5. Unwrap the phase of the TCF along the time axis, t . This step is done to remove artificial discontinuities in the cross-terms region due to the phase being periodic in 2π . Unwrapping the phase, also, reduces the apparent random spikes in the phase due to noise.
6. Apply a median filter to the phase of the TCF along the time axis, t , of length five. This step is done to reduce the noise effects prior to differentiating, since differentiating accentuates these effects due to noise.
7. Differentiate the phase information along the time axis, t . This step changes the unwrapped phase of the TCF from a ramp function to a pseudo-pulse.

8. Apply another median filter along the time axis, t , of length 25. The length of 25 has proven to work well with the T_{hop_min} chosen for our simulations which will be described below. This step is done to again remove the effects due to noise which were accentuated by the differentiation operation in step seven.
9. Calculate a *discrete wavelet transform* along the time axis, t (i.e. for each lag, τ , of the TCF) using the Haar wavelet. This step is done to detect the discontinuities at the edges of the cross-terms region.
10. Sum the wavelet coefficients of the first two scales, d_1 and d_2 of the DWT. Since each successive scale is down-sampled by two when performing a DWT, this step is accomplished by setting $d_2(2n+1) = d_2(2n)$ for $n = 0, 1, 2, \dots, N-1$ where $N = \text{length}(d_2)$, before performing the summation.
11. Perform a $45^\circ/135^\circ$ summation across all values of lag, τ , to obtain a detection vector which has time as its index. This step will be explained further in Subsection V.C.6.
12. Threshold the data in the resulting detection vector obtained in step 11. Further details regarding this threshold scheme are presented in Appendix B and Subsection V.C.7.
13. If the thresholded detection vector contains peaks, the maximum peak value time index represents the estimated hopping time.

With the detection and estimation algorithm given above, we will use the remainder of this chapter to describe the simulation process, apply the algorithm to an example frame, and, finally, to provide the simulation results.

B. SIMULATION

Simulations were conducted to test the effectiveness of the detection and estimation algorithm given above. The Matlab source code for conducting the simulations is given in Appendix A. Five hundred trial experiments were conducted in six different signal to noise ratios (SNR) ranging from 15 dB down to -3 dB. The basic idea behind the experiments was to simulate signals that had already been segmented, as specified in steps one and two of the algorithm. The problem then becomes to determine: a) whether or not a frequency hop exists within the given frame; b) to estimate the hopping time when a frequency hop is detected.

1. Assumptions

As stated in the introduction, it was desired to make as few assumptions as possible on the nature of the frequency hopping signal. With this goal in mind the assumptions were limited to three. The first assumption is of a frequency range within which the spread spectrum signal remains. For the simulations conducted, this range was assumed to be from 1 MHz to 24 MHz. The second assumption was of a minimum hopping time, $T_{\text{hop_min}}$, which was chosen to be 256 sample points. At a sampling rate of 50 MHz, this translates into a minimum hopping time of 5.12 μs . The third, and final,

assumption was on a minimum frequency differential, Δf , for the hop which was chosen to be 1 KHz.

2. Signal Generation

The signals were generated by first choosing a random hopping time, T_{hop} , between sample point 26 and 231 of the 256 point sample frame. The first and last 25 points were disallowed as candidates due to problems with edge effects. In practice, an overlapping scheme could be used for full coverage of the signal. If it is determined that a hop will occur within a simulation frame (i.e. $T_{\text{hop}} \neq 0$), then both first and second frequencies (i.e. frequencies before and after the hop) are randomly generated. The result is a signal with, at most, one hop which can be from any frequency, f_1 , to any frequency, f_2 , such that $1 \text{ MHz} \leq f_1, f_2 \leq 24 \text{ MHz}$, and $|f_1 - f_2| \geq \Delta f$.

The SNR, in decibels, is defined in the simulations as:

$$SNR = 10 \log_{10} \left(\frac{p_{\text{signal}}}{\sigma_{\text{noise}}^2} \right), \quad (5.1)$$

where σ denotes the standard deviation squared, or variance, of the noise, and p_{signal} is the signal power. Forcing the signal to be of unit amplitude, Equation 5.1 becomes:

$$SNR = 10 \log_{10} \left(\frac{1/2}{\sigma_{\text{noise}}^2} \right), \quad (5.2)$$

which allows the calculation of the standard deviation of the additive white Gaussian noise as:

$$\sigma_{noise} = \sqrt{\frac{1}{2} \cdot 10^{\frac{-SNR}{10}}} . \quad (5.3)$$

C. APPLICATION OF DETECTION AND ESTIMATION ALGORITHM TO A SAMPLE FRAME

1. Choosing a Sample Frame

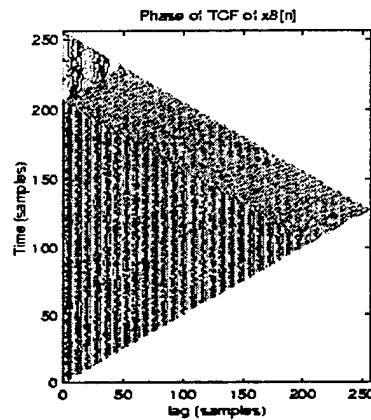
We choose the eighth frame of the signal defined earlier in Section II-B, for the example. This frame is 256 sample points long from sample 1793 to sample 2048, as the frequency hopping signal is segmented into frames of size T_{hop_min} equal to 256 points. Therefore, the eighth frame starts at point $256 \cdot 7 + 1 = 1793$, and ends at point $256 \cdot 8 = 2048$. The hopping time for this frame is at $40 \mu s$, which corresponds to point 2000 for a sampling frequency of 50 MHz. Thus, the hopping time is located at time sample $2000 - 1792 = 208$ inside the frame. The hopping time within our sample frame is called T_{hop} . The frequencies on either side of the hop are $f_{1250} = 6.250$ MHz and $f_{17727} = 22.727$ MHz. Finally, white Gaussian noise is added to the signal to obtain a SNR equal to 10 dB.

2. The Temporal Correlation Function

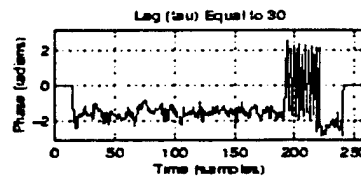
Figure 5.1 plots the phase of the TCF expression for the analytical function derived from the communication signal. Recall that the bottom triangle is due to the auto-terms at frequency f_{1250} and the top triangle is due to the auto-terms at frequency f_{17727} , while the rest are cross-terms due to both frequencies. Figure 5.1b shows the values of the TCF phase for a constant lag equal to 30, τ_{30} .

3. Extracting Phase Information from the TCF

Steps four and five of the detection and estimation algorithm manipulate the phase information of the TCF into a convenient form using the preprocessing tools introduced in Chapter IV. Figure 5.2 plots the unwrapped phase of the TCF shown earlier in Figure 5.1. Figure 5.2b shows the effect of unwrapping the phase at lag, τ_{30} .

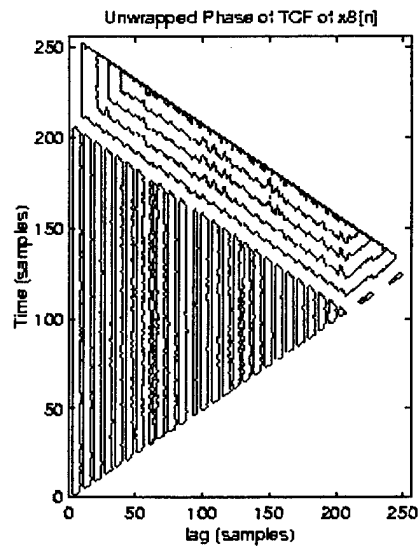


(a)

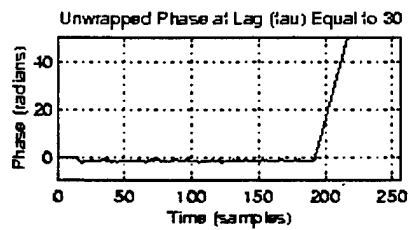


(b)

Figure 5.1: The (a) phase of the Temporal Correlation Function computed on an analytic frequency hopping signal in noise and (b) a closer view at the constant value of lag equal to 30.



(a)



(b)

Figure 5.2: The unwrapped phase information of the TCF of Figure 5.1.

4. Constructing the Pulse

Steps six to eight in the detection and estimation algorithm transform the phase information into a pulse-like signal, or pseudo-pulse. Note that the phase information is usually very sensitive to noise degradation. Thus, several steps are added to minimize noise effects. We first apply a median filter of length 5 to minimize the effects due to

noise while preserving the global trend of the phase information. Next, we differentiate the phase along the time axis t to obtain the pseudo-pulse. Finally, we apply a second median filter of length 25 to again minimize noise effects while preserving the pulse-like shape of the function. Figure 5.3 plots the resulting signal obtained from the TCF phase for the lag time τ_{30} . We now have constructed a pseudo-pulse having the width of the cross-terms and, roughly, centered on T_{hop} . A few comments can be made;

- when the signal is noise free, such processing leads to a perfect pulse of a height equal to the slope of the unwrapped TCF phase, a width equal to the width of cross-terms, and centered on T_{hop} (i.e. centered at sample point 208 of the time axis).
- discontinuities at each edge of the pseudo-pulse are preserved when the signal is filtered with the median filter of length 25.

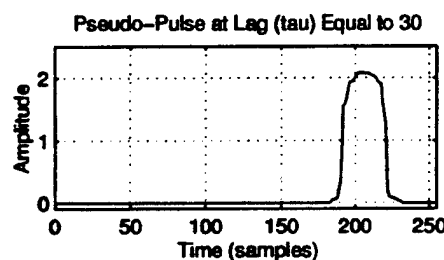


Figure 5.3: The pseudo-pulse formed by differentiating and median filtering the signal of Figure 5.2b.

5. Detecting Discontinuities with the DWT

Next, steps nine and ten, of our detection and estimation algorithm are applied to the pseudo-pulse function. Step nine computes the *discrete wavelet transform* using the Haar wavelet. The transform is well matched to detect discontinuities at the leading and trailing edges of the pseudo-pulse. Next, the wavelet coefficients obtained for the first two scales are averaged to minimize the noise degradation, and to enhance the probability of detecting the discontinuities, as shown earlier in Subsection V.D.4. Figure 5.4 plots

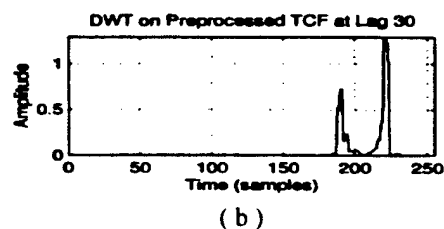
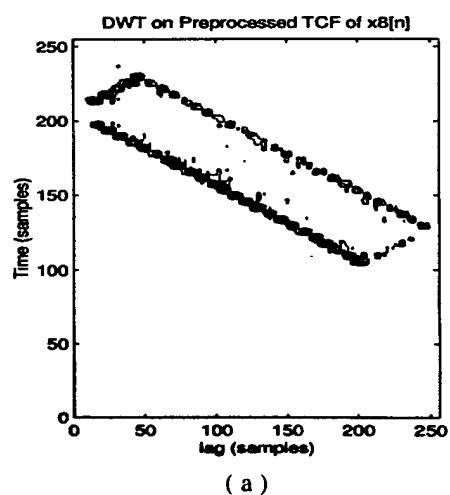


Figure 5.4: DWT coefficients computed for each value of τ (a) on the preprocessed phase of the TCF(t, τ), and (b) for $\tau = 30$.

the resulting wavelet transform obtained from the TCF phase information on the entire TCF, and a detailed view of the transform obtained for lag τ_{30} .

6. Constructing the Detection Vector

Step 11 of our algorithm constructs what will be called the detection vector by performing a $45^\circ/135^\circ$ summation on the DWT matrix. The Matlab code for this summation can be found in Appendix A. However, the basic idea is to sum all the values which represent the edges of the cross-terms in the TCF at 45° and 135° , so that they reinforce each other only at T_{hop} , which we recall is located at point 208 in this example. Figure 5.5a shows pictorially the effects of this summation. The arrows in Figure 5.5b show the directions of summation arrows pointing upward representing 45° and the

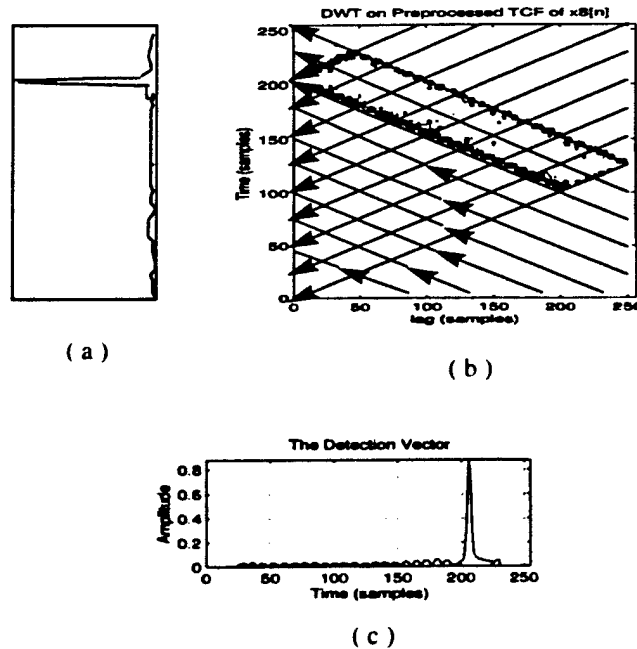


Figure 5.5: Detection vector constructed by performing a $45^\circ/135^\circ$ summation across the columns of the signal shown in Figure 5.4a. (a) shows a representation of the effects of the summation shown in (b) where the arrows indicate the direction of summation. (c) shows a plot of the actual detection vector.

arrows pointing downward representing 135° . The actual detection vector is plotted in Figure 5.5c. Note that the peak indeed appears to very close to point 208.

7. Threshold Determination

Once the detection vector has been formed a decision must be made as to whether or not a hop has occurred within the frame. The statistics of the empirical data from the experiments suggested that the variance of the detection vector would be the best indicator of whether or not a hop had occurred. As a result, the threshold, $T_{threshold}$, is chosen as a multiple, k , of the variance of the detection vector when no hop has occurred within the frame as:

$$T_{threshold} = k \cdot \text{var}(\text{detection vector}_{no_hop}(t)). \quad (5.4)$$

The threshold determination was also guided by the fact that the cost associated with the probability of a missed detection, $P_m = [1 - \text{probability of detection } (P_D)]$ far outweighs the cost associated with the probability of a false alarm, P_{FA} , as the hopping time estimation is only the first step in a complete frequency hopping signal detection scheme. Note that once the hopping times are estimated, the signal frequencies need to be extracted to demodulate the actual message. This step can easily be done by applying frequency analysis to the estimated hopping intervals. Thus, in the case of false alarms, frequency analysis would show the same frequencies in two, or more, consecutive hopping intervals, resulting in no message degradation. However, a missed hopping time will result in degradations in the frequency estimation step, and errors in the decoded message.

Receiver operating characteristics (ROC) curves were generated for each of the six SNR levels and an appropriate threshold chosen for each. This curve, along with the plots of P_D vs. $T_{threshold}$ and P_{FA} vs. $T_{threshold}$, is shown in Figure 5.6 for the SNR = 10 dB case. Similar plots for the other SNR levels are shown in Appendix B. When the SNR = 10 dB, the threshold is chosen to be $k=30$ times the variance of the detection vector generated in a “no hop” frame, which leads to $T_{threshold} = 1.8907 \times 10^{-6}$. Simulations show that the detection vector variance is equal to 0.0076, which exceeds the selected threshold, $T_{threshold} = 1.8907 \times 10^{-6}$. Therefore, a hop is detected. The value of the hop time is estimated next by locating the time index of the detection vector maximum value. This point is 207, which results in an error of one time sample, or a percentage error equal to $\frac{1}{256} \cdot 100\% = 0.39\%$ of T_{hop_min} or approximately 20 ns.

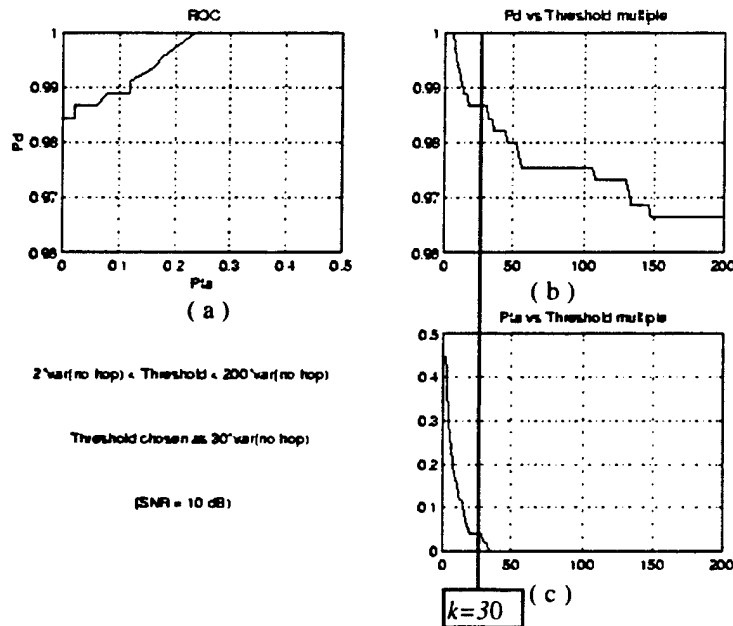


Figure 5.6: Signal in noise at SNR= 10 dB, (a) the ROC curve, (b) P_D vs. $T_{threshold}$, and (c) P_{FA} vs. $T_{threshold}$. Threshold, $T_{threshold}$, chosen as 30 times the variance of a detection vector generated in a “no hop” frame.

D. RESULTS

1. Detection

The detection results given in Table 5.1 show the probability of detection, P_D , the probability of false alarm, P_{FA} , and the percentage of errors in classification for the selected threshold, $T_{threshold}$, at each of the six SNRs considered. Note the entries under the column labeled “ k ” represents a multiple of the variance of the detection vector generated from a “no hop” frame for each respective SNR level. The column labeled “% Error” shows the percentage of misclassifications (i.e., the percentage of false alarms plus misses). Note, also, that the a low probability of false alarm was sacrificed for higher probabilities of detection for reasons discussed earlier. For example, the entries in the row for SNR = 3 dB show that if a 11.4 % misclassification rate and a $P_{FA}=0.1961$ can be tolerated, then we can expect to detect 89.53% of the hops in a given frequency hopping signal.

SNR	k	P_d	P_{fa}	% Error
15 dB	140	1	0	0.0
10 dB	30	0.9866	0.0196	1.4
6 dB	15	0.9844	0.1569	3.0
3 dB	11	0.8953	0.1961	11.4
0 dB	1	0.8129	0.3529	20.4
-3 dB	3	0.6927	0.3333	31.0

Table 5.1: Detection statistics of 500 experiments applying the detection and estimation algorithm of this chapter.

2. Estimation

Simulation results described in Section V-B are given in Table 5.2. The column labeled “Avg. Error” gives the **average** error obtained at each SNR level. For example, at the SNR level of 3 dB, out of all the hops which were detected, the average distance from the true hopping time was 10.48 sample points. This value equates to 4.1% of the minimum hopping time T_{hop_min} . Columns with numeric headings indicate the hop detection probability within a given percentage of T_{hop_min} . For example, at the SNR level of 3 dB, the column labeled “1%” indicates that 36% of all *detected* hops were located within 1% of T_{hop_min} or within 2 points of the true hop time, T_{hop} . Similarly, 72% of all detected hops were located within 5% of T_{hop_min} or within 12 points of the true hop time, T_{hop} . Figure 5.7 plots the distribution of the hopping time detections for all SNR levels considered. Note that as the SNR decreases the distribution spreads out indicating less and less accuracy in the estimation, which is to be expected.

SNR (dB)	1%	5%	10%	15%	20%	30%	40%	50%	75%	100%	Avg. Error (# of samples)
15	0.790	0.984	0.992	0.996	0.996	0.998	1.00	1.00	1.00	1.00	2.22
10	0.726	0.964	0.974	0.978	0.982	0.986	0.986	0.986	0.986	0.986	2.70
6	0.558	0.888	0.926	0.940	0.950	0.960	0.968	0.970	0.970	0.970	5.46
3	0.360	0.720	0.758	0.794	0.828	0.862	0.874	0.882	0.886	0.886	10.48
0	0.116	0.302	0.418	0.510	0.572	0.684	0.752	0.768	0.796	0.796	28.48
-3	0.090	0.174	0.276	0.382	0.456	0.568	0.614	0.646	0.686	0.690	30.99

Table 5.2: Estimation statistics for the 500 experiments at each SNR level using the detection and estimation algorithm described in this chapter. Columns with numeric headings, show the probability that estimated hops are found within a given distance, expressed in percentage of T_{hop_min} , from true hopping times.

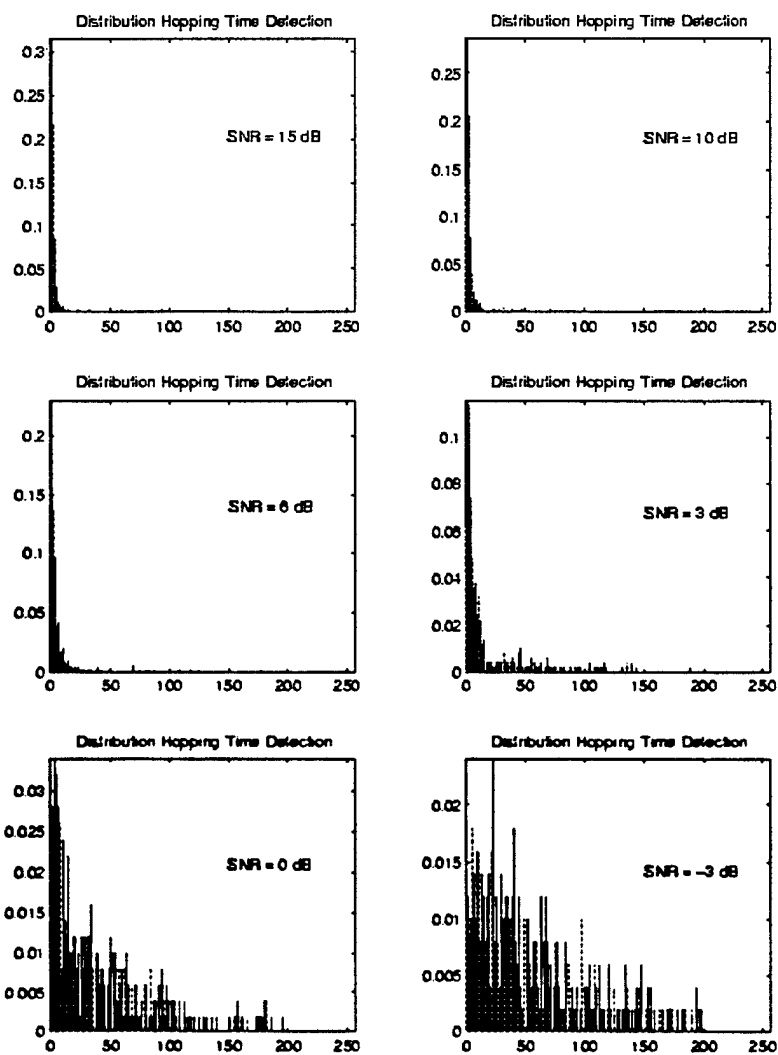


Figure 5.7: The distribution of hopping detections for selected values of SNR.

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

This thesis considered the application of correlation functions and wavelet analysis to the detection and estimation of frequency hopping signals in additive white Gaussian noise. First, we described frequency hopping signals and briefly explained how they are used in spread spectrum communications techniques. Second, we introduced wavelet transforms and showed how they can be used to detect discontinuities in functions and their derivatives. Third, we introduced preprocessing techniques designed to improve the robustness of the detection and estimation scheme in noisy environments. Finally, the detection and estimation algorithm was presented and simulation results shown.

The algorithm developed has only two restrictive assumptions:

1. a minimum hopping time;
2. a minimum frequency differential.

Thus, it can find applications where the minimum hopping time is not held constant; i.e., in *time hopping* signals and hybrid techniques involving either *frequency hopping* or *time hopping*. Results show that the algorithm developed produces acceptable results for SNR levels as low as 3 dB. At this SNR level the percentage of misclassifications is 11.4% and the average error is 4.1% of the minimum hopping time T_{hop_min} .

B. RECOMMENDATIONS

Although the algorithm developed in this work succeeded in meeting its goals, it is believed that improvements could be obtained by altering the approach in two specific areas. First, the *temporal correlation function* produces a two-dimensional "image." However, the algorithm developed here chose to perform one-dimensional wavelet analysis for each individual value of lag, τ . Thus, a possible extension involves considering the problem as an image processing or a pattern recognition problem, due to the specific triangular pattern produced by calculating the TCF of *frequency hopping* signals. As a result, applying two-dimensional edge detection schemes, such as wavelet-based techniques, are expected to improve the robustness of the detection and estimation algorithm. In addition, one could use *neural networks* to recognize the triangular patterns of the TCF calculated for *frequency hopping* signals. Both extensions are presently under consideration.

APPENDIX A. MATLAB SOURCE CODE

The Matlab source code for the detection and estimation algorithm developed in chapter five is provided in this appendix. Functions referenced, but not provided here are either a part of basic Matlab or a part of the *Wavelet Toolbox* [16].

A. SIMULATION PROGRAM

```
% This MATLAB code runs simulations of detection and
% estimation of frequency hopping signals. The simulations
% have been designed to conduct 500 experiments at each of
% 7 noise levels as listed in the SNR_db vector below.
%
% Uses DETECT2V, ERRLOC2

% Filename:      d_esimul.m
%
% Capt Howard Overdyk, last revised 970902

% *****

% Create required signals:

clear

load seed;           % Using set seed only to ensure results are
rand('seed',seed); % reproducible.

delta_f=1000;         % 1 KHz
T_min=256;            % T_min = 256 pts = 5.12 us
f_min=10^6;           % 1 MHz
f_max=24*10^6;        % 24 MHz
fs=50*10^6;           % Sampling frequency = 50 MHz
Nf=(f_max-f_min)/delta_f; % Number of different random freqs fk
Nc=500;               % Number of experiments

% Produce random hop times, Th.

Th=round(206*rand(1,Nc)); % 206 to allow for removing 25 pts from
                        % each end. 0 => no hop
                        % In practice, overlapping technique
                        % would be employed to account for
                        % removing edges.

for i2=0:49,

    Th(1+i2*10)=0;       % ensure 10% are no hops to fully test
                        % detection portion of algorithm
```

```

end

rand('seed',seed);

% Produce random frequencies, Fh, where 1st row is hop from freq.
% and the 2nd row is the hop to freq.

Fh=(round(Nf*rand(2,Ne)).*delta_f)+f_min;

hops=zeros(1,Ne);           % to keep track of if there is a hop and where.
                             % zero (0) => no hop or else it will be hop pt.

x=[];                       % Initialization purposes

% Use random hop times, Th, and random freqs, Fh, to produce signals
% for 500, Ne, experiments.

for i1=1:Ne,

    if Th(i1) == 0,

        n=1:T_min;
        x=[x cos(2*pi*Fh(1,i1)/fs*n)];

    elseif Fh(1,i1) == Fh(2,i1),

        n=1:T_min;
        x=[x cos(2*pi*Fh(1,i1)/fs*n)];

    else

        hops(i1)=Th(i1)+25;    % RN btwn 26 and 231
        n1=1:hops(i1);         % for first freq
        n2=1:T_min-hops(i1);   % for second freq
        x=[x cos(2*pi*Fh(1,i1)/fs*n1) cos(2*pi*Fh(2,i1)/fs*n2)];

    end    % if statement

end        % loop

% *****
%
%      Create Noise and embed signal in it and make analytic.

SNR_db=[15 10 6 3 0 -3]; % SNR in dB
Thresh=[140 30 15 11 1 3]; % Thresholds pre-determined from ROC
                             % function

sigma=sqrt(10.^(-SNR_db(1)/10)/2) ...

```

```

sqrt(10.^(-SNR_db(2)/10)./2) ...
sqrt(10.^(-SNR_db(3)/10)./2) ...
sqrt(10.^(-SNR_db(4)/10)./2) ...
sqrt(10.^(-SNR_db(5)/10)./2) ...
sqrt(10.^(-SNR_db(6)/10)./2)];

randn('seed',seed);
w1=randn(6,length(x));

for i=1:6,

    w(i,:)=sigma(i).*w1(i,:);
    xn(i,:)=hilbert(x+w(i,:));

end

xn(7,:)=hilbert(x);           % No noise case

%*****
%
%    Perform Experiments

for i1=1:7,

    for i2=1:Ne,

        [d(i1,i2),d(i1+7,i2)]=detect2v(xn(i1,(i2-1)*T_min+1:i2*T_min));

    end

    Threshld=Thresh(i1)*d(i1+7,1);   % First experiment is known
                                     % to be a no hop case
    ind1=d(i1+7,:)>Threshld; % Create an indexing vector
                              % Which equals one only if the
                              % variance of the detection
                              % vector exceeds the pre-
                              % determined threshold.
    da=d(i1,:).*ind1;           % All detections which do not
                              % exceed the threshold are set
                              % to a miss

%*****
%
%    Analysis of Results call errloc2

[delta_th(i1,:),Pd(i1),Pfa(i1),pcnt_err(i1),avg_err(i1)]=errloc2(da,hops,Fh);

```


end

B. DETECTION FUNCTION

```
function [t,v]=detect2v(x)
% [T,V]=DETECT2V(X) given signal X, detects the hop time of 256
% samples of a signal X and returns the hop point in T and
% the var(WT summed coeffs) in V.
%
% Uses TCF3, TRDET3U3, SUMLAGS3
%
% Filename:      detect2v.m
%
% Capt Howard Overdyk, last revised 970813
%*****

min_lag=0;
max_lag=length(x)/2;

Rxt=tcf3(x,min_lag,max_lag);

c=trdet3u3(Rxt,min_lag,max_lag);
[ca1,ca2,ca]=sumlags3(c);

cs=[0 ca(1:length(ca)-1)]+[ca(2:length(ca)) 0];

v=var(cs);

[m,t]=max(cs);
```

C. TEMPORAL CORRELATION FUNCTION

```
function Rx=tcf3(x, arg2,arg3)
% TCF3(X,ARG2,ARG3) returns the temporal correlation function(TCF)
% of X. If arg2 is present the tcf will only be calculated
% from min_lag=arg2 out to max_lag=arg3, otherwise the whole
% TCF will be computed for positive values of lag, tau.
%
% Filename:      tcf3.m
%
% Capt Howard Overdyk, last revised 970512
%*****
%
```

```

if nargin>1
    min_lag=arg2;
    max_lag=arg3;
else
    min_lag=0;
    max_lag=length(x)/2;
end

Rx=zeros(length(x)+1,length(x)/2+1);

for lag=min_lag:max_lag;
    for i=lag+1:length(x)-lag,
        Rx(i,lag+1)=x(i+lag)*conj(x(i-lag));
    end
end

%*****
% Compute the TCF for negative values of lag, tau.

%Rx(:,1:length(x)/2)=conj(fliplr(Rx(:,length(x)/2+...
%    2:length(x)+1)));

```

D. WAVELET TRANSFORM OF TCF FUNCTION

```

function ca=trdet3u3(R,min_lag,max_lag)
% TRDET3U3(R,MIN_LAG,MAX_LAG) performs wavelet transforms, using
% Haar wavelet, on all lags from MIN_LAG to MAX_LAG on phase
% of temporal correlation fcn, R, and then sums the
% coefficients of the first two scales of each transform.
% Returns the matrix of summed coefficients CA.
%
% Uses UNWRAP, ANGLE, DIFF, MEDFILT, WLOOKNP

% Filename: trdet3u3.m
%
% Capt Howard Overdyk, last revised 970420

%*****
% Extract unwrapped angle of TCF.

Rxa=unwrap(angle(R));

```

```

%*****
%
%      Investigate different lags of TCF of complex signal to
% determine hop time

level=4;

ca=zeros(129,length(R(:,1))-1);

for lag=min_lag:max_lag,

    % median filter Rxa with length 5, then differentiate
    % and finally, median filter with length 25 before
    % performing the wavelet transforms.

    Rxtmf=medfilt(diff(medfilt(Rxa(:,lag+1).',lag,5)),lag,25);
    c=wtnp(Rxtmf,'db1',level); % Haar == db1

    ca(lag+1,:)=sum(abs(c(1:2,:)));    % sum the 1st two scales

end

```

E. MEDIAN FILTER FUNCTION

```

function xmf=medfilt(x,lag,f_size)
% MEDFIL(X) given data X, which LAG the data came from, and
%      filter size, F_SIZE, this function returns the median
%      filtered data in the vector XMF.
%      LAG is used to determine how large of area outside of
%      the TCF is zeros and so, the function can set these
%      values to the median of the first three values within
%      the TCF.
%
%      Uses MEDIAN

%      Filename:      medfilt.m
%
%      Capt Howard Overdyk, last revised 970421

%*****
%

if nargin < 3

    f_size=3;

end

for i2=1:lag,

```

```

        x(i2)=median(x(lag+1:lag+3));
        x(length(x)-i2+1)=median(x(length(x)-lag-2:length(x)-lag));

    end

    for i= 1:ceil(f_size/2),

        xmf(i)=median(x(1:f_size));

    end

    for i=ceil(f_size/2)+1:length(x)-ceil(f_size/2),

        xmf(i)=median(x(i-ceil(f_size/2)+1:i+floor(f_size/2)));

    end

    for i=length(x)-ceil(f_size/2)+1:length(x),

        xmf(i)=median(x(length(x)-f_size+1:length(x)));

    end

```

F. FOUR LEVEL WAVELET TRANSFORM FUNCTION

```

function m=wtnp(x,wavfamn,level)
% WTNP(X,WAVFAMN,LEVEL) given signal, X,
%     WAVFAMN, the wavelet name (e.g. 'db2'), and
%     the dyadic LEVEL for decomposition, the function
%     returns the wavelet transform coefficients in
%     matrix, M.
%
%     Level <= 4
%
%     Uses COEFMAT4, WAVEDEC

%     Filename:      wtnp.m
%
%     Capt Howard F. Overdyk, last revised 2/7/97

% *****
%

[c,l]=wavedec(x,level,wavfamn);

m=coefmat4(c,l,level,wavfamn);

```

G. WAVELET TRANSFORM COEFFICIENT MATRIX FUNCTION

```

function data=coefmat4(c,l,level,arg3,arg4)
% COEFMAT4(C,L,LEVEL,arg3,arg4) returns an level x Ns matrix of
% the wavelet coefficients (details only) Unnormalized
% coefficients using the DWT tiling and
% using either a specific wavelet ('wname' see WFILTERS)
% or specific wavelet decomposition filters.
% Ns is the length of the signal X
%
% The structure is organized as:
% C = [app. coef.(N)|det. coef.(N)|... |det. coef.(1)]
% L(1) = length of app. coef.(N)
% L(i) = length of det. coef.(N-i+2) for i = 2,...,N+1
% L(N+2) = length(X).
%
% Level <= 4
%
% Uses DETCOEF, WFILTERS, WKEEP

% Filename:      coefmat4.m
%
% Howard F. Overdyk last modified 970807

% *****
%
% Check arguments

if nargin==4
    [LoF_D,HiF_D]=wfilters(arg3,'d');
else
    LoF_D = arg3;  HiF_D = arg4;
end

% Extract approximation and details

N=l(length(l)).

% Expand d1
d1n=detcoef(c,l,1);
d1=wkeep(d1n,N/2^l,'1');

i1=1:2:N; i2=2:2:N;
init=zeros(1,N);
dd1(i1)=d1;dd1(i2)=d1;

% Expand d2
if (level >= 2)

```

```

d2n=detcoef(c,l,2);
d2=wkeep(d2n,N/2^2,'l');
i1=1:4:N; i2=2:4:N; i3=3:4:N; i4=4:4:N;
dd2=init;
dd2(i1)=d2;dd2(i2)=d2;dd2(i3)=d2;dd2(i4)=d2;
end

% Expand d3
if (level >= 3)
d3n=detcoef(c,l,3);
d3=wkeep(d3n,N/2^3,'l');
i1=1:8:N; i2=2:8:N; i3=3:8:N; i4=4:8:N;
i5=5:8:N; i6=6:8:N; i7=7:8:N; i8=8:8:N;
dd3=init;
dd3(i1)=d3;dd3(i2)=d3;dd3(i3)=d3;dd3(i4)=d3;
dd3(i5)=d3;dd3(i6)=d3;dd3(i7)=d3;dd3(i8)=d3;
end

% Expand d4
if (level >= 4)
d4n=detcoef(c,l,4);
d4=wkeep(d4n,N/2^4,'l');
i1=1:16:N; i2=2:16:N; i3=3:16:N; i4=4:16:N;
i5=5:16:N; i6=6:16:N; i7=7:16:N; i8=8:16:N;
i9=9:16:N; i10=10:16:N; i11=11:16:N; i12=12:16:N;
i13=13:16:N; i14=14:16:N; i15=15:16:N; i16=16:16:N;
dd4=init;
dd4(i1)=d4;dd4(i2)=d4;dd4(i3)=d4;dd4(i4)=d4;
dd4(i5)=d4;dd4(i6)=d4;dd4(i7)=d4;dd4(i8)=d4;
dd4(i9)=d4;dd4(i10)=d4;dd4(i11)=d4;dd4(i12)=d4;
dd4(i13)=d4;dd4(i14)=d4;dd4(i15)=d4;dd4(i16)=d4;
end

```

```
data=[dd1;dd2;dd3;dd4];
```

H. 45° MATRIX SUMMATION FUNCTION

```
function [ca1,ca2, ca]=sumlags3(c)
```

```

% SUMLAGS3(C) performs a 45 degree summation over all columns of the
% wavelet transform coefficients, C, in both directions
% of 45 deg and returns a row vector CA with the result.
% CA1 contains results of summing only in 45 deg direction.
% CA2 contains results of summing only in 135 deg direction.
% Edges are clipped to avoid edge effects using EFILT2.
%
% Uses results of EDGEFILT2

% Filename:      sumlags3.m
%
% Capt Howard Overdyk, last revised 970707

```

```

%*****
%

load efilt2;          % see code of edgefilt2.m
[m,n]=size(c);
c=efilt2.*c;          % Remove edges of WT coeffs.
ca1=zeros(1,n);
ca2=zeros(1,n);

for i1 = 1:n-m+1, % covers area where we have all rows to
    % sum over at 45 degrees

        for i2 = 1:m,

            ca1(i1)=ca1(i1)+c(i2,i1-1+i2);

        end
    end

    ca1=ca1./m;      % Normalize summation

    c1=1;

    for i1 = n-m+2:n, % covers end where we aren't summing over
        % all rows at 45 degrees.

            for i2 = 1:m-c1,

                ca1(i1)=ca1(i1)+c(i2,i1-1+i2)./(m-c1);

            end
            c1=c1+1;

        end

    end

    for i1 = n:-1:m, % covers area where we have all rows at 135.

        for i2 = m:-1:1,

            ca2(i1)=ca2(i1)+c(i2,i1-i2+1);

        end
    end

    ca2=ca2./m;      % Normalize summation

    c1=1;

```

```

for i1 = m-1:-1:1, % covers area where we don't have all rows
    % at 135 degrees.

```

```

    for i2 = m-c1:-1:1,

```

```

        ca2(i1)=ca2(i1)+c(i2,i1-i2+1)./(m-c1);

```

```

    end

```

```

    c1=c1+1;

```

```

end

```

```

ca=ca1+ca2;

```

I. EDGE FILTER CREATION PROGRAM

```

% EGDEFILT creates a matrix which will filter off the edge effects
% of a wavelet transform of a TCF of size 129 x 256.

```

```

% Filename:      edgefilt2.m

```

```

%

```

```

% Capt Howard Overdyk, last revised 970707

```

```

% *****

```

```

clear

```

```

cutoff=25;

```

```

m=129;

```

```

n=256;

```

```

m1=128-cutoff;

```

```

f1=zeros(m1,m1);

```

```

for i1=1:m1,

```

```

    f1=f1+diag(ones(1,i1),(i1-m1));

```

```

end

```

```

f2=[fliplr(f1) f1];

```

```

f2=flipud(f2);

```

```

f=[f2; zeros(26,2*m1)];

```

```

efilt2=[zeros(m,25) f zeros(m,25)];

```

```

save /home/dsp0b/overdyk/matlab/thesis/fq_wk7/efilt2.mat efilt2;

```

J. DETECTION AND ESTIMATION ANALYSIS FUNCTION

```

function [delta_th,Pd,Pfa,pcnt_err,avg_err]=errloc2(d,hops)

```



```

% [DELTA_TH,Pd,Pfa,PCNT_ERR,AVG_ERR]=ERRLOC2(D,HOPS) will
%   given:  D      - a 1x500 vector containing the location of the
%              detected hops for each experiment.
%           HOPS    - a 1x500 vector containing the actual location
%              of the hops for each experiment.
%
%   returns:
%           DELTA_TH- a 1x257 vector containing the distribution
%              of correctly detected hops versus the
%              distance from the actual hop time in #
%              of samples.
%           PD      - a scalar value of the probability of detection
%           PFA     - a scalar value of the probability of a
%              false alarm.
%           PCNT_ERR- a scalar value of the percentage of mis-
%              classifications.
%           AVG_ERR  - a scalar value indicating, of the actual
%              detections, what the average distance, in
%              # of samples, from the true hop time was.
%
%   side effects:
%           Produces two plots of the distribution of the location
%           of the detections relative to the actual hopping time
%           once in line graph format and then in bar graph format.
%
%   Filename:      errloc2.m
%
%   Capt Howard Overdyk, last revised 970805

% *****
%
%   Analysis of Results

Ne=500;
Tot_hops2=Ne;
Ntcf=256;

N=Ne*Ntcf;
max_dth=Ntcf;
dth_ind=0:max_dth;
delta_th=zeros(1,max_dth+1);
delta_m=zeros(1,max_dth+1);
loc_ind=0:Ntcf;
loc=zeros(1,257);
f_of_m=zeros(2,Ne);           % first two rows correspond to freq either
                               % side of missed hops on experiment # = col #

% Initialize output variables

Pd=0;
Pfa=Pd;

```

```

dets=Pd;
fa=Pd;
miss=Pd;
nn=Pd;

i1=1;                                % remains constant

    for i2=1:Ne,                      % the number of experiments

        acth=hops(i2);

        exph=d(i1,i2);                % hops detected by experiment

        delta_h=abs(acth-exph);

        if acth ~= 0,                 % hop exists

            if exph ~= 0,              % found hop

                delta_th(i1,delta_h+1)=delta_th(i1,delta_h+1)+1;
                delta_m(i1,i2)=delta_h;
                dets(i1)=dets(i1)+1;

            else                        % missed detection

                loc(i1,acth)=loc(i1,acth) + 1;
                miss(i1) = miss(i1) + 1;
            end                        % found hop if statement

        else                           % no hop exists case

            if exph ~= 0,              % false alarm case

                fa(i1) = fa(i1) + 1;

            else                        % no hop no detect case

                delta_th(i1,delta_h+1)=delta_th(i1,delta_h+1)+1;
                delta_m(i1,i2)=delta_h;
                % delta_h always equals zero (0) here.
                nn(i1) = nn(i1)+1;

            end                        % false alarm if statement

        end                            % hop exists if statement

    end                                % Inner for loop i2

% end                                  % Outer for loop i1

```

```
figure
orient tall
```

```
[i3,i4,v]=find(hops~=0);
```

```
% Print relevant statistics to MATLAB session
```

```
Tot_hops=sum(v)
```

```
dets
```

```
miss
```

```
fa
```

```
nn
```

```
Pd=dets/Tot_hops
```

```
Pfa=fa/(Ne-Tot_hops)
```

```
subplot(221)
```

```
plot(dth_ind,delta_th(1:)/Tot_hops2,'m-');
```

```
xlabel('delta Th'),ylabel('Detections')
```

```
title('Distribution Hopping Time Detection')
```

```
% print out in tabular form the percentage of detections
```

```
% within 1, 5, 10, 15, 20, 30, 40,50,75, and 100 percent
```

```
% of the minimum hop time.
```

```
p_tothop=sum(delta_th.)/Tot_hops2
```

```
p_tot1=sum(delta_th(:,1:3).)/Tot_hops2
```

```
p_tot5=sum(delta_th(:,1:13).)/Tot_hops2
```

```
p_tot10=sum(delta_th(:,1:26).)/Tot_hops2
```

```
p_tot15=sum(delta_th(:,1:39).)/Tot_hops2
```

```
p_tot20=sum(delta_th(:,1:52).)/Tot_hops2
```

```
p_tot30=sum(delta_th(:,1:77).)/Tot_hops2
```

```
p_tot40=sum(delta_th(:,1:103).)/Tot_hops2
```

```
p_tot50=sum(delta_th(:,1:129).)/Tot_hops2
```

```
p_tot75=sum(delta_th(:,1:193).)/Tot_hops2
```

```
p_tot100=sum(delta_th.)/Tot_hops2
```

```
subplot(222)
```

```
bar(0:max_dth,delta_th(1:)/Tot_hops2);
```

```
axis([0,max_dth,-inf,inf])
```

```
title('Distribution Hopping Time Detection')
```

```
avg_err=mean(delta_m)
```

```
pcnt_err=(fa + miss)/Ne
```

K. RECEIVER OPERATING CURVES GENERATION FUNCTION

```

function [Pd,Pfa]=roc(d,hops, inc_size)
% [Pd,Pfa]=ROC(D,HOPS,INC_SIZE) given a vector, D, of detected hops
%      a vector, HOPS, of actual hops, and INC_SIZE which is the
%      amount to increase the threshold by at each increment, will
%      return a probability of detection vector, PD, and a
%      probability of false alarm vector, PFA.

%      Filename:      roc.m
%
%      Capt Howard Overdyk, last revised 970913

%*****
%
%      Analysis of Results

Ne=500;
Tot_hops2=Ne;
Ntcf=256;

N=Ne*Ntcf;
max_dth=Ntcf;
dth_ind=0:max_dth;
delta_th=zeros(1,max_dth+1);

% Initialize output variables

Pd=zeros(1,100);
Pfa=Pd;
dets=Pd;
fa=Pd;
miss=Pd;
nn=Pd;

Th=d(2,1)

[i3,i4,v]=find(hops~=0);
Tot_hops=sum(v)

for i1=1:100;

    TTh=inc_size*i1*Th;
    ind1=d(2,:)>TTh;
    d1=d(1,:).*ind1;

    for i2=1:Ne,                % the number of experiments

        acth=hops(i2);
    
```

```

        exph=d1(i2);                % hops detected by experiment
        delta_h=abs(acth-exph);

        if acth ~= 0,                % hop exists
            if exph ~= 0,            % found hop
                dets(i1)=dets(i1)+1;

            else                      % missed detection
                miss(i1) = miss(i1) + 1;
            end                      % found hop if statement
        else                          % no hop exists case
            if exph ~= 0,            % false alarm case
                fa(i1) = fa(i1) + 1;

            else                      % no hop no detect case
                % delta_h always equals zero (0) here.
                nn(i1) = nn(i1)+1;

            end                      % false alarm if statement
        end                          % hop exists if statement

    end                              % Inner for loop i2

end                                  % Outer for loop i1

Pd=dets/Tot_hops;
Pfa=fa/(Ne-Tot_hops);

figure
subplot(221)
plot(Pfa,Pd)
xlabel('Pfa'),ylabel('Pd')
title('ROC')
grid on

k=inc_size*[1:100];
subplot(222)
plot(k,Pd)
title('Pd vs Threshold multiple')
grid on

```

```
subplot(224)
plot(k,Pfa)
title('Pfa vs Threshold multiple')
grid on
```


APPENDIX B. THRESHOLD DETERMINATION

Figures B.1 to B.5 plot the curves used to select detection threshold values for SNR levels of 15 dB, 6 dB, 3 dB, 0 dB, and -3 dB, respectively. The threshold, $T_{threshold}$, is given by:

$$T_{threshold} = k \cdot \text{var}(\text{detection vector}_{no_hop}(t)), \quad (\text{B.1})$$

where k is the multiple which needs to be determined. The horizontal axis in plots (b) and (c) of Figures B.1 to B.5 represents the parameter k . Therefore, the selection of k is based on choosing an acceptable P_D level or an acceptable P_{FA} level. The parameter, k , is chosen so that it leads to a high probability of detection, P_D , and an acceptable P_{FA} level. This selection procedure is explained in further detail in Section V.C.7.

For example, let us consider the 6 dB case plotted in Figure B.2. Let us assume that we need $P_D = 0.985$. Figure B.2b shows that the corresponding required value of k equals 15. This value of k results in a P_{FA} level equal to 0.16. Note, that these values of P_D and P_{FA} are located at a point very near the “elbow” of the ROC curve shown in Figure B.2a.

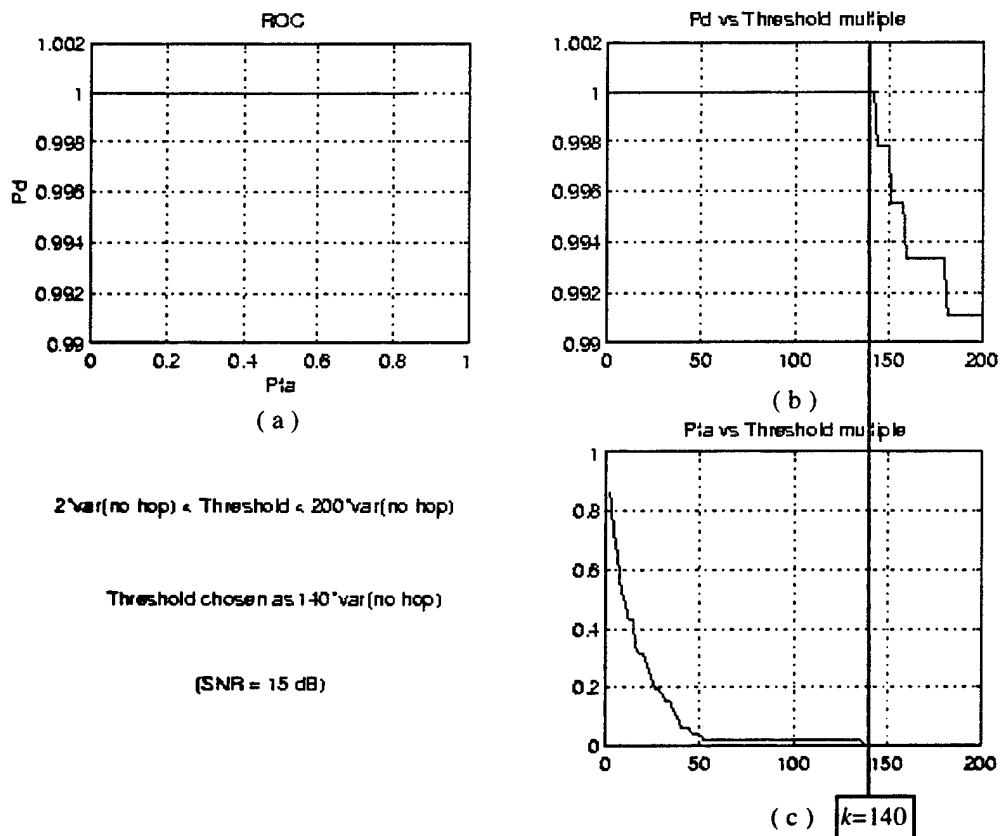


Figure B.1: Noisy signal with SNR = 15 dB, (a) ROC curve, (b) P_D vs. $T_{\text{threshold}}$, and (c) P_{FA} vs. $T_{\text{threshold}}$. Threshold, $T_{\text{threshold}}$, chosen as 140 times the variance of the detection vector generated in a "no hop" frame.

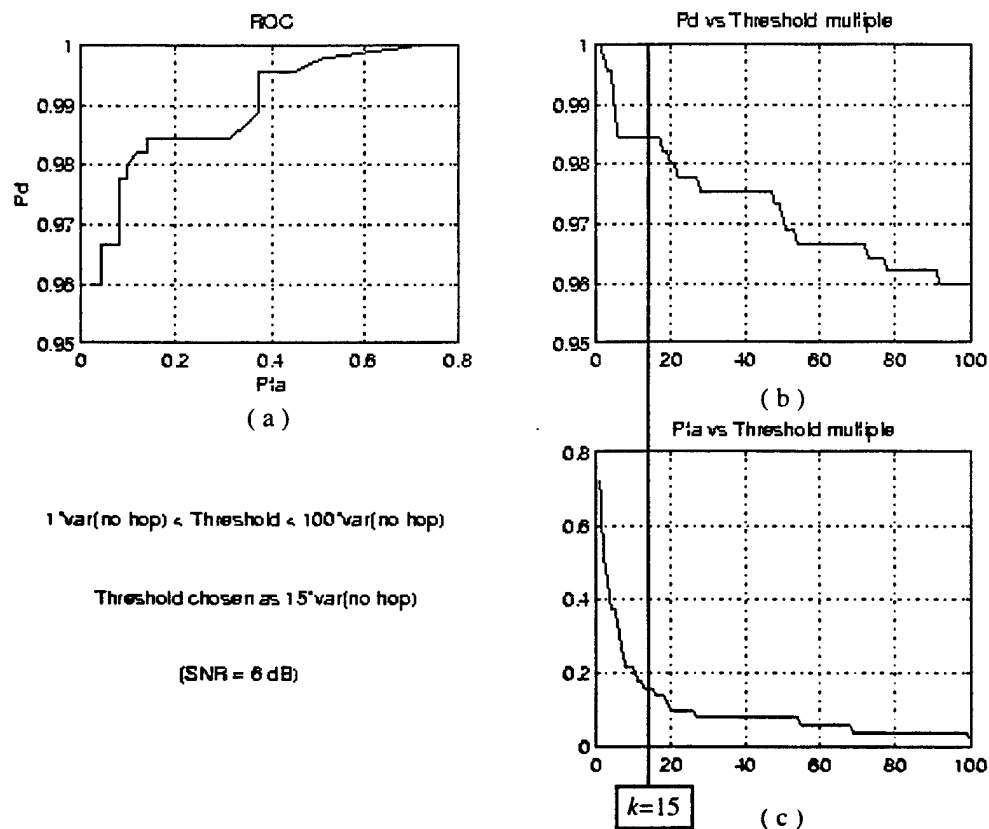


Figure B.2: Noisy signal with SNR = 6 dB, (a) ROC curve, (b) P_D vs. $T_{\text{threshold}}$, and (c) P_{FA} vs. $T_{\text{threshold}}$. Threshold, $T_{\text{threshold}}$, chosen as 15 times the variance of the detection vector generated in a "no hop" frame.

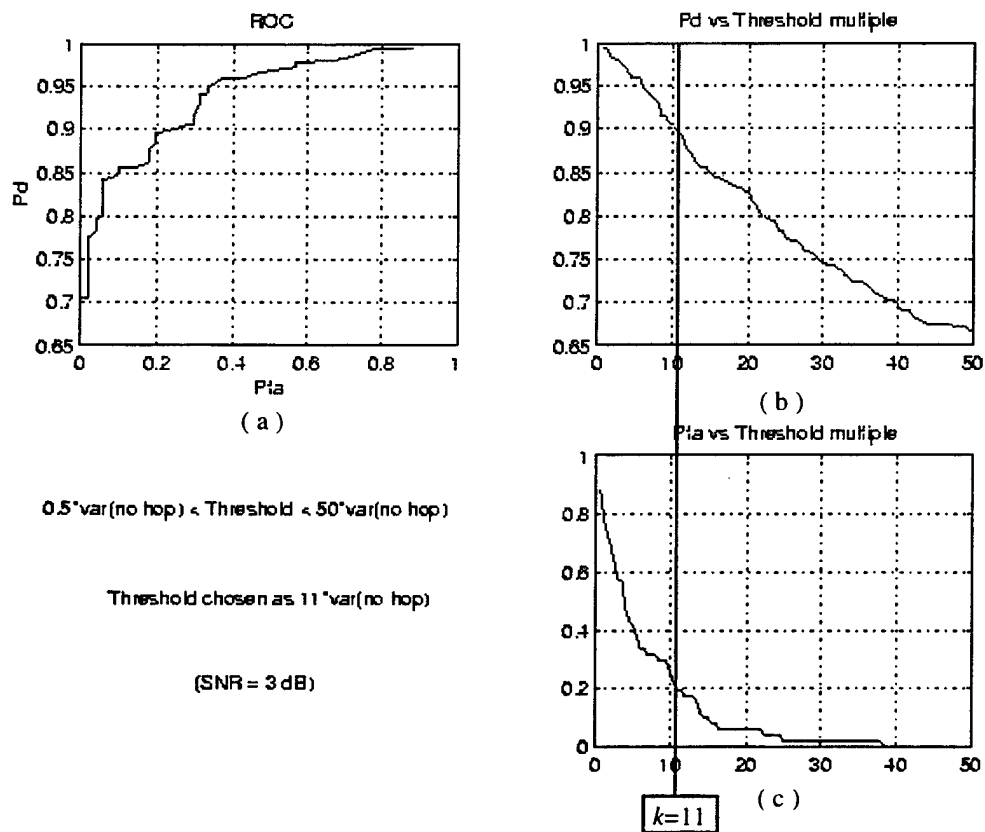


Figure B.3: Noisy signal with SNR = 3 dB, (a) ROC curve, (b) P_D vs. $T_{\text{threshold}}$, and (c) P_{FA} vs. $T_{\text{threshold}}$. Threshold, $T_{\text{threshold}}$, chosen as 11 times the variance of the detection vector generated in a "no hop" frame.

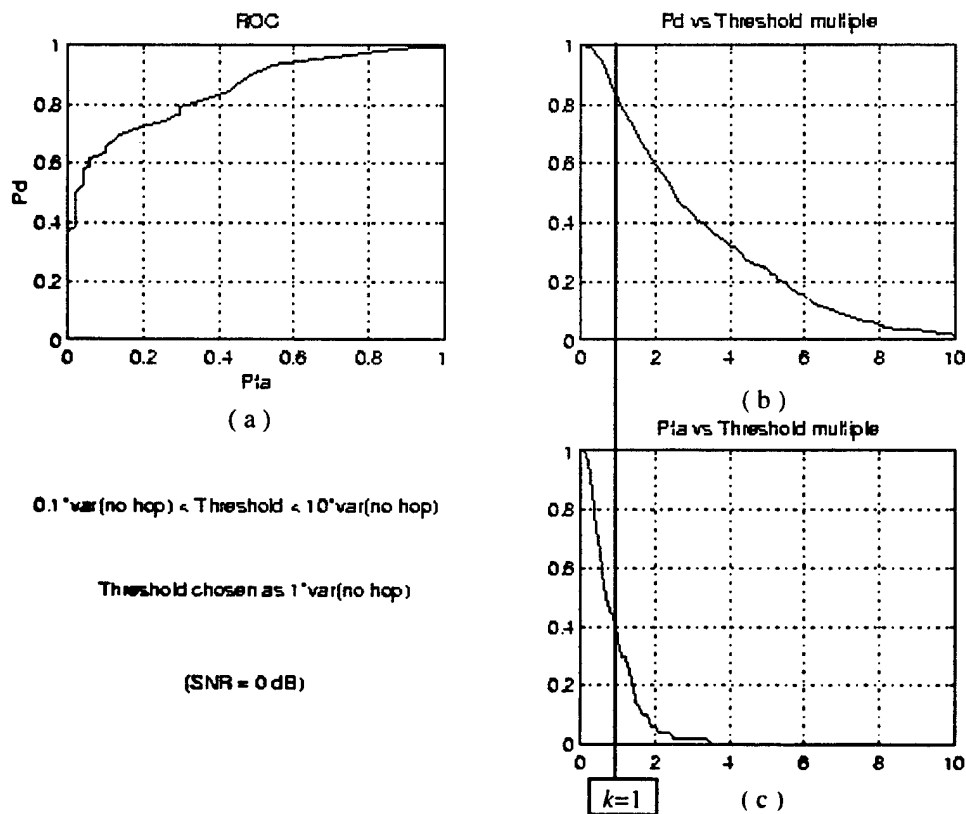


Figure B.4: Noisy signal with SNR = 0 dB, (a) ROC curve, (b) P_D vs. $T_{\text{threshold}}$, and (c) P_{FA} vs. $T_{\text{threshold}}$. Threshold, $T_{\text{threshold}}$, chosen as 1 times the variance of the detection vector generated in a "no hop" frame.

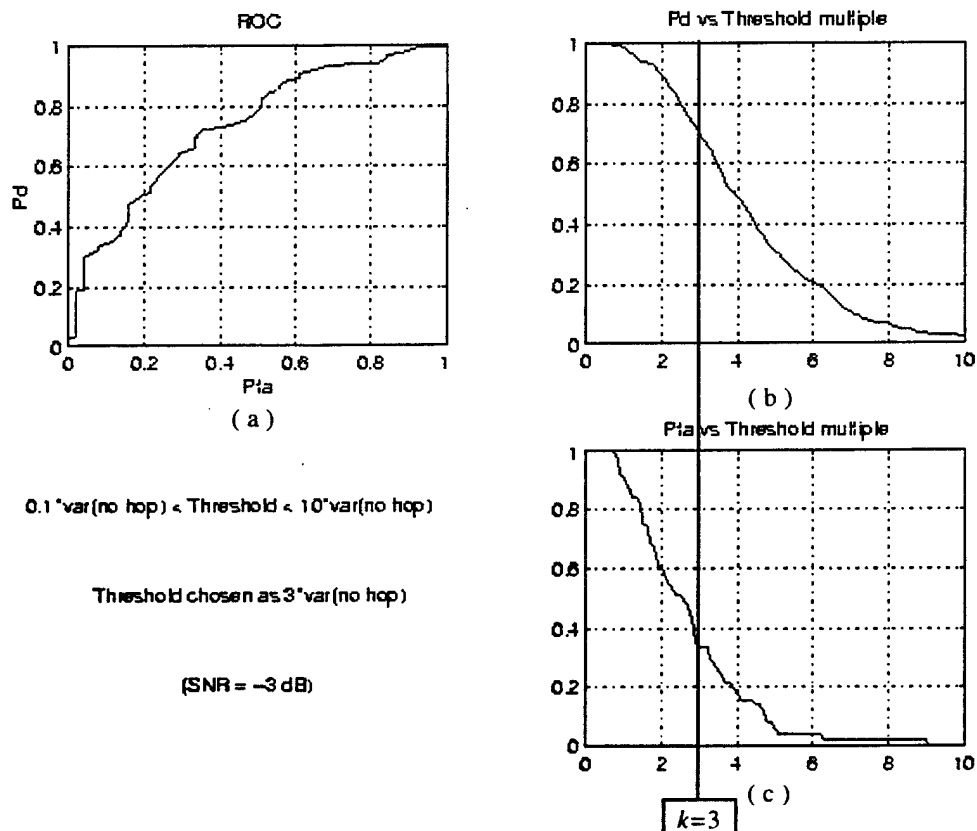


Figure B.5: Noisy signal with SNR = -3 dB, (a) ROC curve, (b) P_D vs. $T_{\text{threshold}}$, and (c) P_{FA} vs. $T_{\text{threshold}}$. Threshold, $T_{\text{threshold}}$, chosen as 3 times the variance of the detection vector generated in a “no hop” frame.

LIST OF REFERENCES

- [1] R. Peterson, R. Ziemer, and D. Borth, *Introduction to Spread Spectrum Communications*, Prentice Hall, Upper Saddle River, New Jersey, 1995.
- [2] N. Beaulieu, W. Hopkins, and P. McLane, "Interception of Frequency-Hopped Spread-Spectrum Signals," *IEEE J. Select. Areas Commun.*, vol. 8, no. 5, pp. 853-870, June 1990.
- [3] L. Aydin and A. Polydoros, "Hop-Timing Estimation for FH Signals Using a Coarsely Channelized Receiver," *IEEE Trans. Commun.*, vol. 44, no. 4, April 1996.
- [4] K. Ho, W. Prokopiw, and Y. Chan, "Modulation Identification by the Wavelet Transform," *Proceedings of MILCOM '95*, San Diego, CA, November 1995.
- [5] M. Simon, U. Cheng, L. Aydin, A. Polydoros, and B. Levitt, "Hop Timing Estimation for Noncoherent Frequency-Hopped M-FSK Intercept Receivers," *IEEE Trans. Commun.*, vol. 43, no. 2/3/4, pp. 1144-1154, February/March/April 1995.
- [6] C. Chung and A. Polydoros, "Parameter Estimation of Random FH Signals Using Autocorrelation Techniques," *IEEE Trans. Commun.*, vol. 40, no. 1, pp. 149-159, January 1992.
- [7] O. Rioul and M. Vetterli, "Wavelets and Signal Processing," *IEEE SP Magazine*, pp 14-38, October 1991.
- [8] B. Sklar, *Digital Communications: Fundamentals and Applications*, PTR Prentice Hall, Englewood Cliffs, New Jersey, 1988.
- [9] C. Shannon, "Communication in the Presence of Noise," *Proc. IRE*, pp. 10-21, January 1949.
- [10] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, Massachusetts, 1996.
- [11] C. Burrus, R. Gopinath, and H. Guo, *Wavelets and Wavelet Transforms: A Primer*, Prentice Hall, Upper Saddle River, New Jersey, 1998.
- [12] A. Oppenheim and A. Willsky, *Signals and Systems*, Prentice Hall, Englewood Cliffs, New Jersey, 1983.
- [13] F. Hlawatsch and G. Boudreaux-Bartels, "Linear and Quadratic Time-Frequency Signal Representations," *IEEE SP Magazine*, pp. 21-67, April 1992.

- [14] The Easton Press, *The Concise Oxford Dictionary*, Norwalk, Connecticut, 1991.
- [15] M. Shensa, "The Discrete Wavelet Transform: Wedding the À Trous and Mallat Algorithms," *IEEE Trans. Signal Processing*, vol. 40, no. 10, October 1992.
- [16] The Mathworks, Inc., *Wavelet Toolbox User's Guide*, Massachusetts, 1996.
- [17] B. Friedlander, High Performance Spectral Analysis Techniques for Non-stationary Signals, Technical Report, Signal Processing Technology Ltd., 22 June 1995.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center2
8725 John J. Kingman Rd., STE 0944
Ft. Belvoir, Virginia 22060-6218
2. Dudley Knox Library2
Naval Postgraduate School
411 Dyer Rd.
Monterey, California 93943-5101
3. Director, Training and Education1
MCCDC, Code C46
1019 Elliot Rd.
Quantico, Virginia 22134-5027
4. Director, Marine Corps Research Center2
MCCDC, Code C40RC
2040 Broadway Street
Quantico, Virginia 22134-5107
5. Director, Studies and Analysis Division1
MCCDC, Code C45
300 Russell Road
Quantico, Virginia 22134-5130
6. Chairman, Code EC1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5121
7. Prof. Monique P. Fargues, Code EC/Fa3
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5121
8. Prof. Ralph Hippenstiel, Code EC/Hi2
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5121

9. Capt Howard F. Overdyk1
415 North 2nd Street
Apt. #206
Winterset, Iowa 50273
10. Marine Corps Representative1
Naval Postgraduate School
Code 037, Bldg, 234, HA-220
699 Dyer Rd.
Monterey, CA 93990
11. Marine Corps Tactical Systems Support Activity.....1
Technical Advisory Branch
Attn: Maj J.C. Cumiskey
Box 555171
Camp Pendleton, CA 92055-5080